ADDIS ABABA UNIVERSITY

ADDIS ABABA INISTITUTE OF TECHNOLOGY

SCHOOL OF CIVIL AND ENVIRONMENTAL ENGINEERING

SOFTWARE DEVELOPMENT FOR THE ANALYSIS OF LATERAL EARTH PRESSURE AND DESIGN OF RETAINING WALLS

BY

ERMIAS TAMIRAT TESFAYE

AUGUST 2015

ADDIS ABABA, ETHIOPIA

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

ADDIS ABABA INISTITUTE OF TECHNOLOGY

SCHOOL OF CIVIL AND ENVIRONMENTAL ENGINEERING

Software Development for the Analysis of Lateral Earth Pressure and Design of Retaining Walls

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for Degree of Master of Science in Civil Engineering
(Geotechnical Engineering)

BY

ERMIAS TAMIRAT TESFAYE

ADVISOR

PROF. ALEMAYEHU TEFERRA

August 2015

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY
Addis Ababa Institute of Technology
School of Civil and Environmental Engineering

Software Development for the Analysis of Lateral Earth Pressure and Design of Retaining Walls

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for Degree of Master of Science in Civil Engineering
(Geotechnical Engineering)

By: Ermias Tamirat Tesfaye

Approved by Board of Examiners:

Prof. Alemayehu Teferra *Advisor*	______________ *Signature*	______________ *Date*
Dr.Ing. Henok Fikre *External Examiner*	______________ *Signature*	______________ *Date*
Dr.Ing. Samuel Tadesse *Internal Examiner*	______________ *Signature*	______________ *Date*
Dr. Esayas G/Youhannes *Chairman*	______________ *Signature*	______________ *Date*

DECLARATION

I, the undersigned, declare that this thesis is my original work performed under the supervision of my research advisor Prof. Alemayehu Teferra and has not been presented as a thesis for a degree in any other university. All sources of materials used for this thesis have also been duly acknowledged.

Name: Ermias Tamirat Tesfaye

Signature __________________

Place: Addis Ababa Institute of Technology,
School of Civil and Environmental Engineering,
Addis Ababa University,
Addis Ababa

Date: August 2015

ACKNOLEDGEMENT

First of all, I would like to thank God who has given me the opportunity, courage and perseverance in life.

I would like to pass my deepest gratitude to Ethiopian Roads Authority (ERA), which has provided me with the opportunity and sponsored me throughout the MSc Programme. Special thanks go to my advisor Professor Alemayehu Teferra whose guidelines and suggestions have added huge value to my work. Moreover, I cannot leave without acknowledging my friend and class mate, Biruk Assefa, Who has provided an invaluable support throughout the programme.

Last but not least, I would like to thank my loving family for their continuous support and encouragement.

ABSTRACT

Analysis of lateral earth pressure is one of the basic tasks that a geotechnical engineer deals with while analyzing or designing earth retaining structures. In the design stage, the lateral earth pressure should be estimated accurately and stability of the retaining structure be checked for different destabilizing factors.

In order to analyze/design the earth retaining structure practicing and academic professionals have to go through tiresome and routine step by step procedures. Usage of commercially available programs to tackle these problems would be an ideal solution. However, such applications are costly and wouldn't be the choice of professionals in developing countries, like Ethiopia.

Different theories have been suggested by different authors to compute the lateral earth pressure that soils exert on retaining structures. Earth pressure theories suggested by Coulomb and Rankine are the most well known and termed as "Classical Earth Pressure Theories". These theories have been considered reliable in spite of the limitations they have and are considered basic to the problem. One of the major drawbacks of the theories is that they assume the failure surface to be plane throughout. As time goes by, these theories have been modified to suit the different actual conditions and come up with better results. Bell (1915) has further developed Rankine's equation to handle cohesive soils. Kerisel and Absi (1990) have developed a chart putting in to account that, the failure plane has got a logarithmic spiral at its bottom portion together with a tangent extending further to the ground.

This thesis aims at developing software for the analysis of lateral earth pressure and design of gravity and cantilever retaining walls for external stability based on the theories suggested by; Coulomb, Rankine, Bell and Kerisel and Absi. The developed software, GeoLEPARD, helps in analyzing earth pressures and getting the right retaining structure dimension fast and simple.

Outputs of the software have been compared against results of hand/spread sheet calculations. The comparison has shown that, with proper data entry, the software can generate similar output enhanced with numeric and graphical displays.

Table of Contents

List of Figures:

List of Tables

1. INTRODUCTION

1.1. Background

These days, the use of software is becoming part of our day to day life. With the increase in computational power and sophistication of computers, more and more software have been developed to solve large and complex problems in a cost-effective and efficient ways.

In many areas of engineering, software application has become indispensable. The reason for the same is due to the vastness and the complexity of practical engineering problems.

This thesis aims at developing software for the analysis of lateral earth pressure and design of gravity and cantilever retaining walls for their external stability.

The problem of determining the lateral pressure against retaining walls is one of the oldest in the field of engineering. Several investigators have proposed many theories of earth pressure after a lot of experimental and theoretical work. Of these theories, those given by Coulomb and Rankine shall be discussed in detail.

Rankine's theory deals with earth pressure in cohesionless granular materials, i.e. assuming that $c' = 0$. However, in 1915, Bell published an extension of Rankine's theory which included the cohesion parameter. In Rankine's theory the slope of the wall back and the development of friction between the soil and the wall are ignored. Thus, calculated values are somewhat pessimistic. Coulomb, however considered the orientation of the back wall and soil friction while computing the lateral earth pressure.

Both in the Rankine and Coulomb earth pressure theories, the potential failure surfaces in the backfill were considered to be planes. In reality most failure surfaces in soil are curved. To take account of the effects of wall friction different authors have assumed the curved failure surface to be a logarithmic spiral, circular arc or an ellipse. In this thesis works of Kerisel and Absi (1990) have been considered to account for curved failure surface.

1.2. Objective of the Thesis

The general objective of this thesis is to develop software that is used for the design and analysis of gravity and cantilever retaining walls.

The specific objectives of the thesis are:

- To enhance the calculation process and to provide a tool for practical purposes.
- To develop a simpler, faster and accurate way of analyzing the lateral earth pressure and design of retaining walls.
- To help both academicians and practitioners to compare results obtained by hand calculations.

1.3. Problem Statement

Hand calculations of earth pressure analysis and retaining wall design are laborious and time consuming. The development of simple software would enhance the calculation process and would serve as a tool for practice, especially in developing countries where licensed software cannot be easily obtained.

1.4. Organization of the Thesis

This thesis work comprises six Chapters. The first Chapter contains a general introduction about the study, the objective of the study and its problem statement.

The second Chapter is dedicated to literature review. In this part, highlights are given concerning previous works conducted regarding lateral earth pressure and retaining walls. Moreover, equations proposed by different authors will be discussed.

In the third Chapter, the methodology of the study is discussed. This includes the works done to arrive at the final result (i.e. the software), the inputs required for the software, the cases considered and detailed procedures required to come up with solution for a specific scenario.

In the fourth Chapter, the procedures to use the software are presented. In the fifth Chapter, the application of the software is presented. In the sixth chapter, which is the last Chapter, the conclusions made from the study and the recommendations for further studies are stated.

2. LITERATURE REVIEW - LATERAL EARTH PRESSURE

2.1. General

Lateral earth pressure is a significant design element in a number of foundation engineering problems. Vertical or near-vertical slopes of soil are supported by retaining walls, cantilever sheet-pile walls, sheet-pile bulkheads, braced cuts, and other similar structures. The earth pressure to which the above types of retaining structures are subjected is commonly referred to as lateral earth pressure.

These structures require a quantitative estimate of the lateral pressure on a structural member for either a design or stability analysis. The proper design of those structures requires an estimation of lateral earth pressure, which is a function of several factors, such as; type and amount of wall movement, shear strength parameters of the soil, unit weight of the soil and drainage conditions in the backfill. Figure 2.1 shows a retaining wall of height H. For similar types of backfill,

a) The wall may be restrained from moving (Figure 2.1a). The lateral earth pressure on the wall at any depth is called the *at-rest earth pressure.*

b) The wall may tilt away from the soil that is retained (Figure 2.1b). With sufficient wall tilt, a soil wedge behind the wall will fail. The lateral pressure for this condition is referred to as *active earth pressure.*

c) The wall may be pushed into the soil that is retained (Figure 2.1c). With sufficient wall movement, a soil wedge will fail. The lateral pressure for this condition is referred to as *passive earth pressure.*

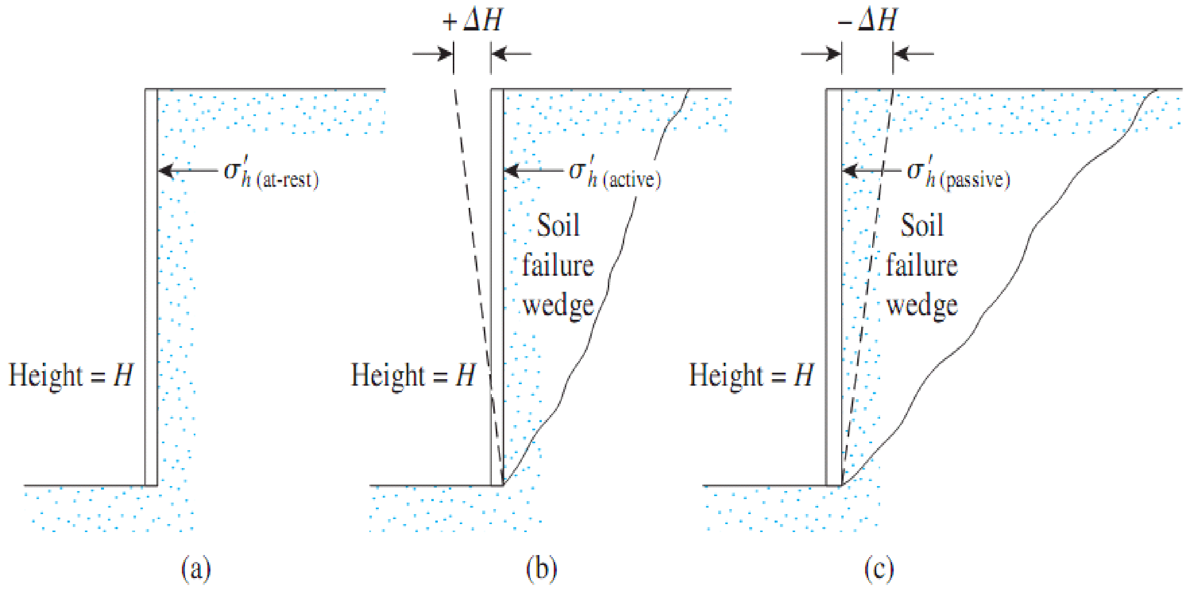


Figure 2.1: Nature of lateral earth pressure on a retaining wall

The method of plastic equilibrium as defined by the Mohr rupture envelope is most generally used for estimating the lateral pressure from earth and other materials such as grain, coal, and ore. On occasion one may use the finite-element (of the elastic continuum) method but this has several distinct disadvantages for most routine design. The FEM has more application for estimating pressure on tunnel liners and large buried conduits than for most lateral pressure analyses.[10]

Earth pressures are developed during soil displacements (or strains) but until the soil is on the verge of failure, as defined by the Mohr's rupture envelope (see Fig. 2.2), the stresses are indeterminate. They are also somewhat indeterminate at rupture since it is difficult to produce a plastic equilibrium state in a soil mass everywhere simultaneously; most times it is a progressive event. Nevertheless, it is common practice to analyze rupture as an ideal state occurrence, both for convenience and from limitations on obtaining the necessary soil parameters with a high degree of reliability.

Referring to Fig. 2.2, one can see the two circles that are common to point A and tangent to the rupture line. Both these circles represent a state of plastic equilibrium in plane strain. One of the other circles such as EA or AF would be a steady-state K_0 condition depending on the overconsolidation ratio (OCR). [10]

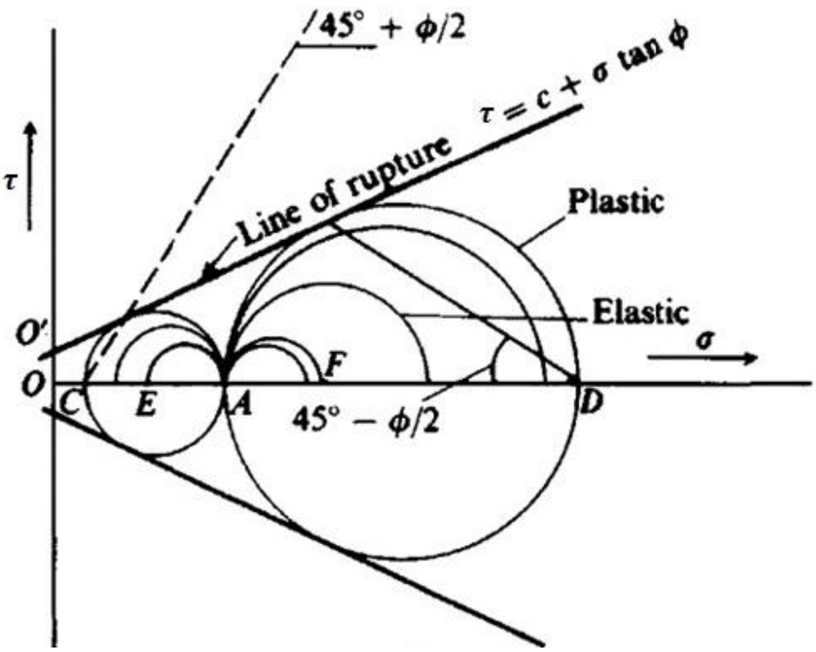


Figure 2.2: Mohr's circles for the K_o and Plastic equilibrium (rupture) [9]

The gradual decrease or increase of pressure on the wall with the movement of the wall from at rest condition may be depicted as shown in Fig. 2.3.

The ratio $\frac{\Delta H}{H}$ required to develop passive state is considerably larger than required for active state.

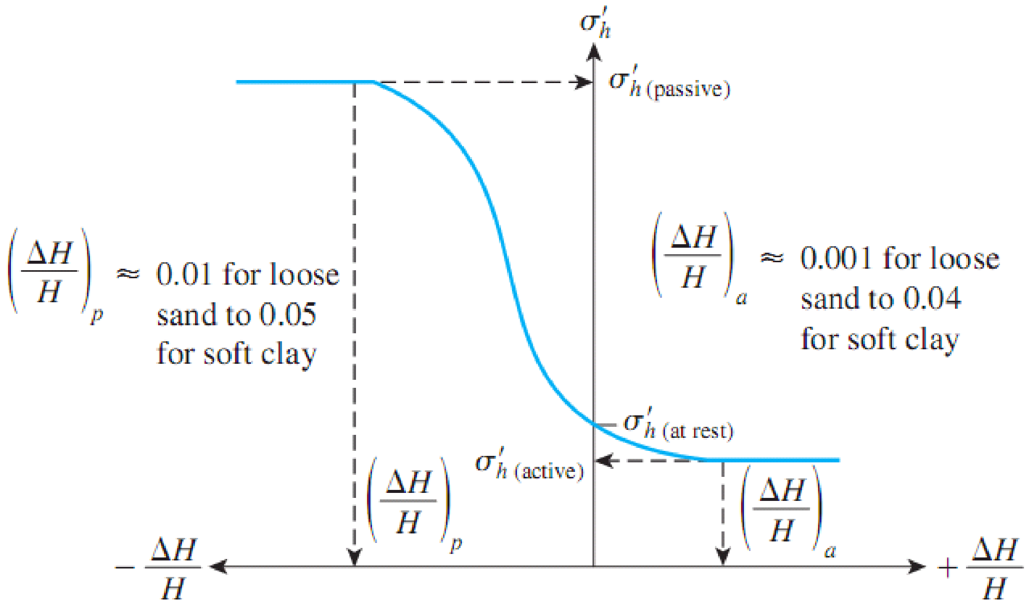


Figure 2.3: Nature of variation of lateral earth pressure at a certain depth [3]

The strain required to mobilize the full active pressure behind a wall will only mobilize a portion of the passive pressure in front of a wall, so the full amount of passive pressure should not be relied on.

Moreover, rigidity of the walls plays a role in mobilization of the active earth pressure. Walls that yield sufficiently can get the full active pressure mobilized. With structures where yielding is restricted, the horizontal pressure acting could be greater than the active pressure and near the “at rest” condition.

Type of wall movements also determines the amount of active and passive pressures mobilized. The active and passive pressures are fully mobilized at relatively smaller movements for rotation about the top and translation than rotations about the toe which requires about 2-3 times as much. [7]

2.2. Earth pressure at rest

Over geological time the stresses in a soil mass at a particular level stabilize into a steady state and strains become zero. When this occurs the vertical and lateral stresses become principal stresses acting on principal planes.

This effective stress state at zero strain is termed the at-rest condition.

As explained in the preceding section, at the at-rest condition the wall does not yield at all (that is, either away from the soil mass or toward the soil mass). Figure 2.4 shows a retaining wall with a horizontal backfill. At a depth z below the ground surface, the vertical and horizontal effective stresses are σ'_h and σ'_v respectively. The ratio of σ'_h to σ'_v is usually referred to as the lateral earth pressure coefficient at rest, or;

$$k_o = \frac{\sigma'_h}{\sigma'_v} \qquad (2.1)$$

Where k_o = earth pressure coefficient at-rest

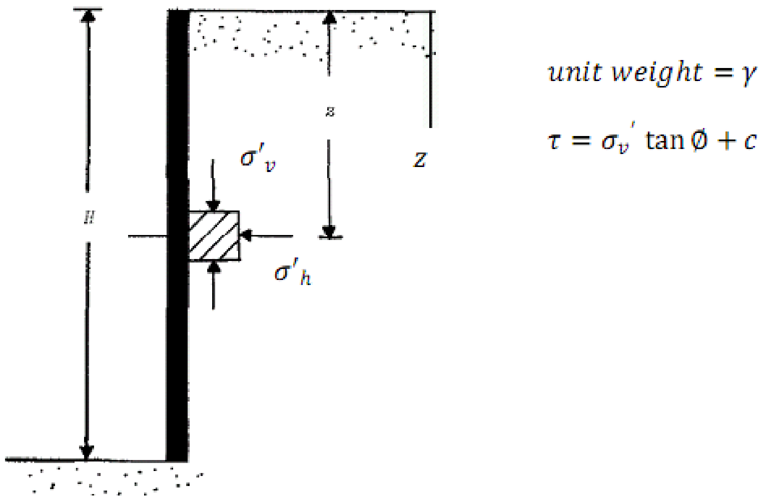


Figure 2.4: Definition of lateral earth pressure at-rest.

The most common and widely used relationship of k_o for design is that given by Jaky (1944), which is of the form;[4]

$$k_o = 1 - \sin \emptyset' \qquad (2.2)$$

Where $\emptyset'$ = drained friction angle

In recent times, it has been shown by several investigators that Eq. (2.1) is applicable only to normally consolidated loose soils. This fact can be explained by referring to Fig. 2.5, which shows a plot of $\sigma_h{}'$ against$\sigma_v{}'$. Along the branch OA (initial loading), the soil is normally consolidated. Mayne and Kulhawy (1982) have analyzed and published results of about 121 soils[4]. Based on their analysis they found that, for both sand and clayey soils, the best fit line from linear regressions analysis can be given as:

$$K_{o(nc)} = 1 - 1.003 \sin \emptyset \qquad (2.3)$$

where K_0 (nc) = at-rest earth pressure coefficient for normally consolidated soils .

So it appears that the original Jaky equation is generally valid for all normally consolidated sands. Brooker and Ireland (1965) have also given the following relationships for normally consolidated clays:[4]

$K_{o(nc)} = 0.4 + 0.007(PI)$ (PI between 0 and 40) (2.4a)

$K_{o(nc)} = 0.64 + 0.001(PI)$ (PI between 40 and 80) (2.4b)

where PI is plasticity index in percent

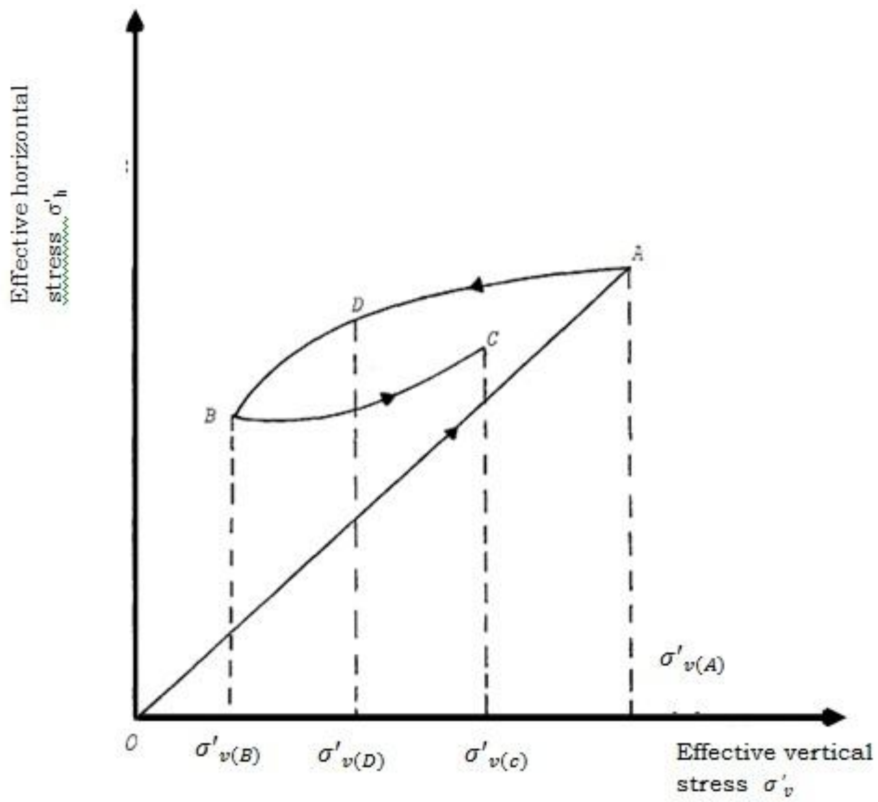


Figure 2.5: Variation of $\boldsymbol{\sigma_h}'$ against $\boldsymbol{\sigma_v}'$ for loading , unloading, and reloading cycles. (NOTE: OA=initial loading, ADB = initial unloading; and BC –reloading branch.)[**4**]

It can be seen however from Fig. 2.5 that at point D, which is on the initial unloading path, the magnitude of K_O is higher than that observed along OA because the soil is overconsolidated.

For overconsolidated soil, the at-rest earth pressure coefficient may be expressed as (Mayne and Kulhawy, 1982):[4]

$$K_o = (1 - \sin \emptyset) OCR^{\sin \emptyset} \quad (2.5)$$

Where, OCR = Overconsolidation Ratio

One can readily derive a value for K_0 in terms of Poisson's ratio based on the definition of K_0 being an effective stress state at zero strain. From Hooke's law the lateral strain in terms of the effective horizontal(x, y) and vertical (z) stresses is;

$$\epsilon_x = 0 = \frac{1}{E_s}\left(\sigma_x - \mu\sigma_z - \mu\sigma_y\right) = \epsilon_y \qquad (2.6)$$

With $\sigma_x = \sigma_y = k_o\sigma_z$ we obtain, on substitution into the preceding and cancelling,

$$k_o = \frac{\mu}{1-\mu} \qquad (2.7)$$

Kezdi (1972) suggests that for sloping ground the k_o relation ships can be modified as follows;[6]

$$k_{o,\beta} = k_o(1 + sin\beta) \qquad (2.8)$$

Where; β is the angle with the horizontal.

2.2.1. At-rest Force and Resultant

According to Eq. (2.1);

$$\sigma'_h = k_0\sigma'_v$$

The preceding equation is valid for effective stress conditions. If ground water is present, the total horizontal stress can be expressed as:

$$\sigma_h = k_o\sigma'_v + u \qquad (2.9)$$

Figure 2.6a shows a retaining wall of height H with the ground water level at a depth $z = z_1$. A surcharge q (per unit area) is also applied at the ground surface. The at-rest force per unit length of the retaining wall can be determined by plotting the variation of σ_h with depth z. Figure 2.6b, c, and d shows three cases which are described below.

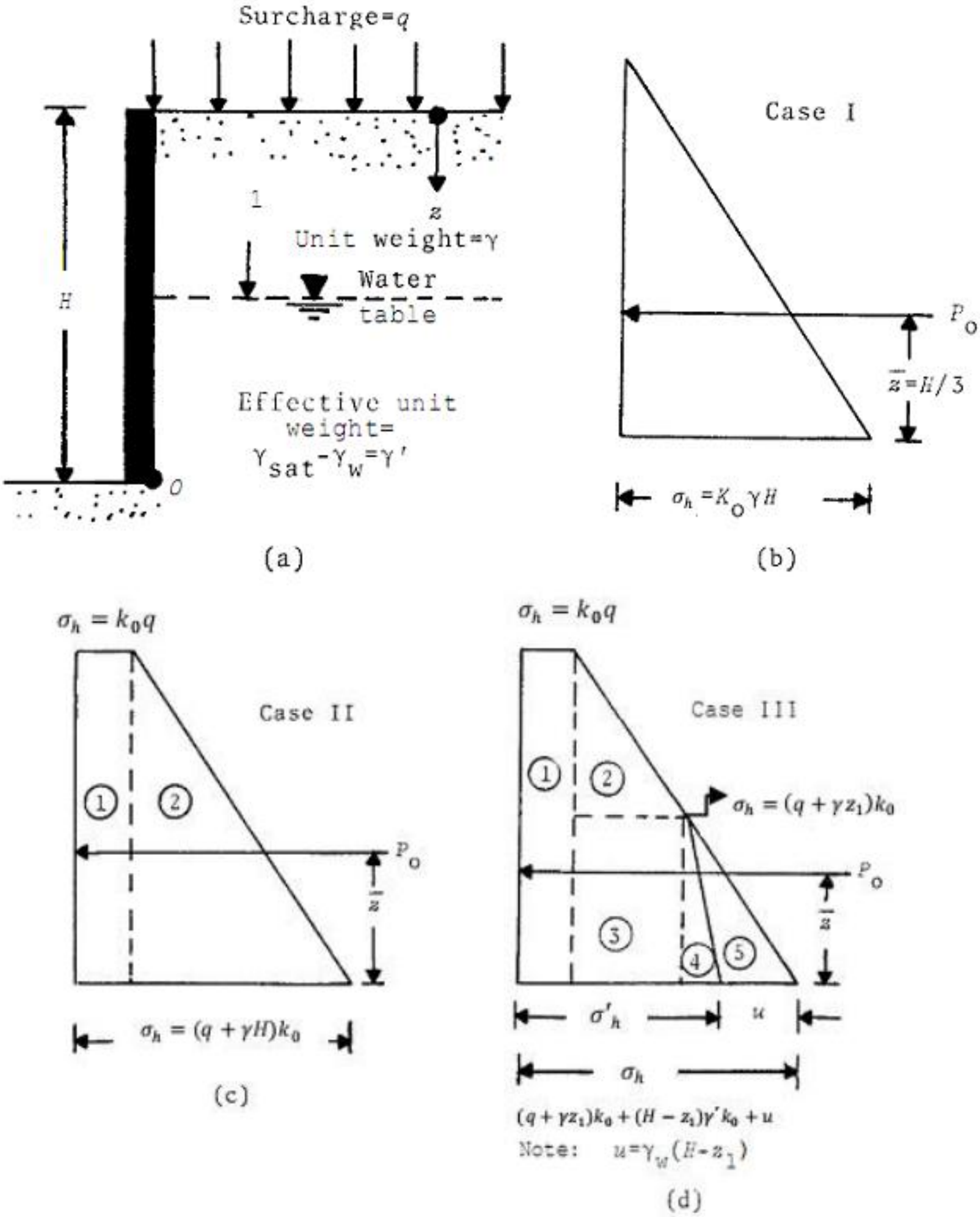


Figure 2.6: At-rest force and resultant[**4**]

Case I--q=O; $z_1 \geq$ H (Fig. 2.6b)

At z=0, $\sigma_h = 0$

At z=H, $\sigma_h = k_o \gamma H$; u = 0

Where γ = unit weight of soil

So, force per unit length;

P_o = area of the pressure diagram = $\frac{1}{2} k_o \gamma H^2$; (2.10)

Case II--q≠0; $z_1 \geq$H (Fig. 2.6c)

P_o = Area 1 + Area 2

$$= k_o qH + \frac{1}{2} k_o \gamma H^2 \quad (2.11)$$

Case III-- q≠0; $z_1 <$H (Fig. 2.6d)

P_o = $\sum Area1 + Area2 + Area3 + Area4 + Area5$

$$= k_o qH + \frac{1}{2} k_o \gamma {z_1}^2 + k_o \gamma z_1 (H - z_1) + \frac{1}{2} k_o \gamma' (H - z_1)^2 + \frac{1}{2} \gamma_w (H - z_1)^2 \quad (2.12)$$

where γ'= effective unit weight of soil below water (γ_{sat} - γ_{wat})

The location of the line of action of the resultant can be determined by taking the moment of the areas about the bottom of the retaining wall, or;

$$\bar{z} = \frac{\Sigma\ Moment\ of\ the\ area\ about\ O}{P_o} \quad (2.13)$$

In certain circumstances, it may be required to determine the at-rest force per unit length of a retaining wall which has an inclined back face as shown in Fig. 2.7. For this case, the at-rest force on the vertical face BC, Po (BC), should be determined first [4], 0r;

$$P_{o(BC)} = \frac{1}{2} k_o \gamma H^2 \quad (2.14)$$

The at-rest force on the back of the wall, Po, can be given as:

$$\bar{P_o} = \bar{P}_{o(BC)} + \bar{W_s} \quad (2.15)$$

Where $\bar{W_s}$ = weight of the soil wedge ABC per unit length of the wall

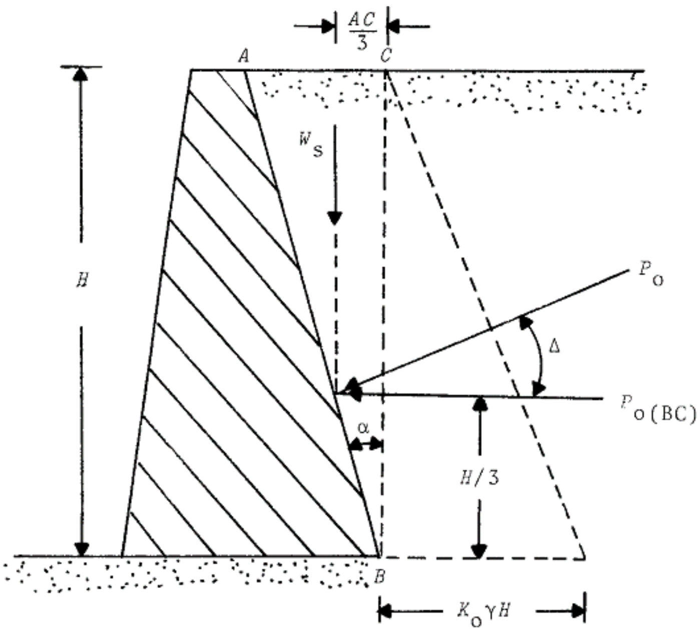


Figure 2.7: At-rest force for a retaining wall with an inclined back face.

$$W_s = \frac{1}{2}\gamma H^2 \tan\alpha \qquad (2.16)$$

Hence, the magnitude of the force would be:

$$P_o = \sqrt{P^2{}_{o(BC)} + W_s{}^2} \;=\; \frac{\gamma H^2}{2}\sqrt{k_o{}^2 + \tan^2\alpha} \qquad (2.17)$$

2.3. General States of Plastic Equilibrium[13]

A body of soil is said to be in a state of plastic equilibrium if every part of it is in an incipient failure condition.

Plastic collapse occurs after the state of equilibrium has been reached in part of a soil mass, resulting in the formation of an unstable mechanism; that part of the soil mass slips relative to the rest of the mass. The applied load system, including body forces, for this condition is referred to as the collapse load. Determination of the collapse load using plasticity theory is complex and would require that the equilibrium equations, the yield criterion and the flow rule were satisfied with in the plastic zone: the compatibility condition would not be involved unless specific deformation condition were imposed. However plasticity theory also provides the means of avoiding complex analyses. The limit theorems of plasticity can be used to calculate the lower and upper bounds to the true collapse load. In certain cases the theorems produce the same result, which would then be the exact value of the collapse load. The limit theorems can be stated as follows;

a) Lower bound theorem

If a state of stress can be found which at no point exceeds the failure criterion for the soil and which is in equilibrium with a system of external loads (which includes the self-weight of the soil), then collapse cannot occur: the external load system thus constitutes a lower bound to the true collapse load (because a more efficient stress distribution may exist).

b) Upper bound theorem

If a mechanism of plastic collapse is postulated and if, in an increment of displacement, the work done by a system of loads is equal to the dissipation of energy by the internal stresses then collapse must occur: the external load system thus constitutes an upper bound to the true collapse load (because a more efficient collapse mechanism may exist).

In the lower bound approach the conditions of equilibrium and yield are satisfied with no consideration of the mode of deformation: the Mohr-Coulomb failure criterion is also taken to be the yield criterion. In the upper bound approach a mechanism of plastic collapse is formed by choosing a slip surface and the work done by the external forces is equated to the loss of energy by the stresses acting along the slip surface, without consideration of equilibrium. The chosen collapse mechanism is not necessarily the true mechanism but it must kinematically be admissible, i.e. the motion of the sliding soil mass must be compatible with its continuity and with any boundary restrictions.

2.4. Earth Pressure Theories

The magnitude of the lateral earth pressure is evaluated by the application of one or the other of the so-called “lateral earth pressure theories” or simply “earth pressure theories”. The problem of determining the lateral pressure against retaining walls is one of the oldest in the field of engineering. Several investigators have proposed many theories of earth pressure after a lot of experimental and theoretical work. Of all these theories, those given by Coulomb and Rankine stood the test of time and are usually referred to as the “Classical earth pressure theories”. These theories have been considered reliable in spite of some limitations and are considered basic to the problem. These theories have originally been developed to apply to cohesionless soil backfill, since this situation is considered to be more frequent in practice and since the designer will be on the safe side by neglecting cohesion. Later researchers gave necessary modifications to take in to account cohesion, surcharge, submergence and others.

In this thesis, these two well-known classical earth pressure theories shall be discussed in detail. Moreover, the work of Kerisel and Absi shall be covered in depth.

2.5. Rankine's Theory of Earth Pressure

Rankine (1857) developed his theory of lateral earth pressure when the backfill consists of dry cohesionless soil. The theory was later extended by Resal (1910) and Bell (1915) to be applicable to cohesive soils.

Rankine's theory presents a solution for a mass of cohesionless soil in a state of limiting equilibrium. The magnitude of the lateral pressure, (σ'_h), depends only on the vertical effective stress and the shear strength of the soil, and thus the problem is statically determinate. No account is taken of displacement and so this is a lower bound solution.

The limiting lateral earth pressures are defined in terms of the effective vertical stress:

The horizontal active earth pressure, $\sigma'_{ha} = k_a \sigma'_v$ (2.18a)

The horizontal passive earth pressure, $\sigma'_{hp} = k_p \sigma'_v$ (2.18b)

Where k_a = the coefficient of active earth pressure
k_p = the coefficient of passive earth pressure

k_a and k_p, may be expressed in terms of the angle of friction $\emptyset'$ as follows;

$$k_a = \frac{\sigma'_{ha}}{\sigma'_v}$$

$$k_a = \frac{1-\sin\emptyset'}{1+\sin\emptyset'} = \tan^2(45 - \emptyset'/2) \quad (2.19a)$$

Similarly, $$k_p = \frac{1+\sin\emptyset'}{1-\sin\emptyset'} = \tan^2(45 + \emptyset'/2) \quad (2.19b)$$

2.5.1. Lateral pressure in cohesionless soils

The simplest set of active earth pressure conditions occurs against a wall having a smooth vertical back, retaining a soil with a horizontal unloaded surface, and with no standing water behind the wall (Fig. 2.9). From Rankine's theory:

At depth z, $\sigma'_{ha}, = k_a\sigma'_v \; = k_a\gamma z$ (2.20)

where γ = bulk unit weight of the soil

At the base of the wall, z=H and $\sigma'_{ha} = k_a\gamma H$

The resultant active thrust (P_A) acting normal to the wall surface is given by:

$$P_A = \int_0^H \sigma'_v dz = \int_0^H k_a\gamma z.dz \qquad (2.21)$$

i.e. the area of the pressure distribution diagram.

For the distribution shown in Fig. 2.9:

$$P_A = \frac{1}{2}k_a\gamma H^2 \qquad (2.22)$$

The line of action of P_A passes through the centre of area at a height of 1/3H above the base.

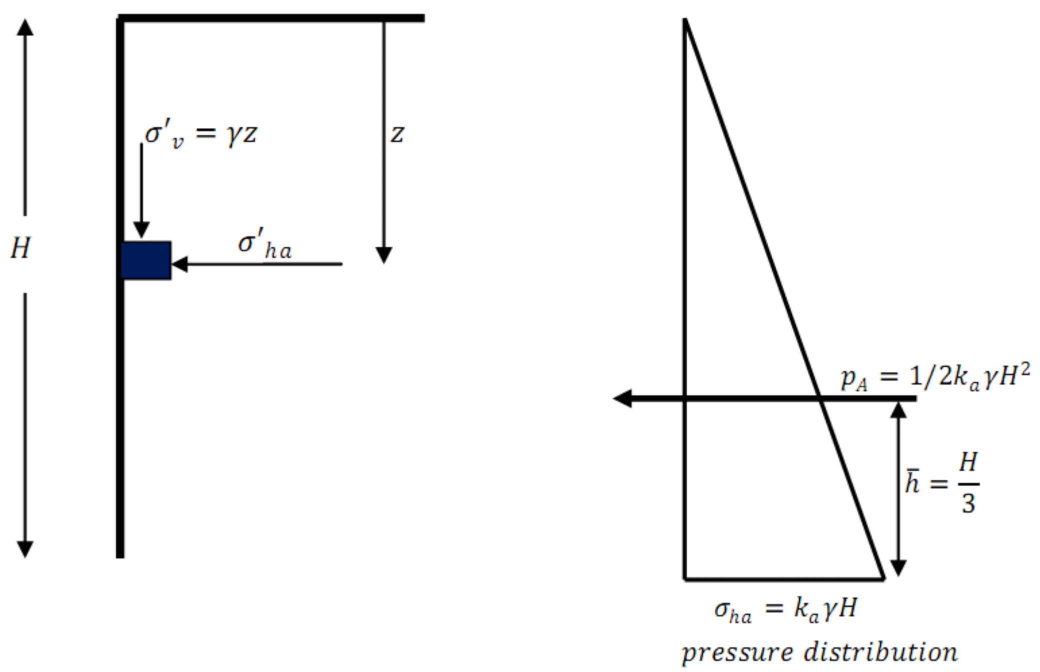
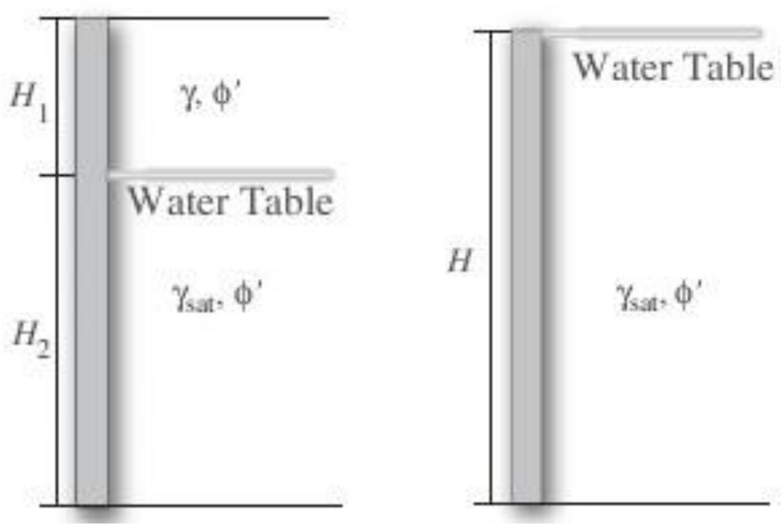


Figure 2.8: Active pressure distribution in cohesionless soil

2.5.1.1. Effect of groundwater level

Retaining walls should be provided with adequate back-face drainage. When the retained soil itself is not free-draining, a drainage blanket of coarse gravel and/or geotextile layers may be laid, often with a collecting drain behind the heel of the wall. Wherever practicable, the groundwater level behind the wall should not be allowed to rise above the base. If, as a result of poor drainage, drain blockage or accidental flooding, the water level rises behind the wall, the vertical effective stresses will be reduced, but the horizontal total stresses will be increased. If the retained soil becomes fully waterlogged (i.e. the groundwater level is at the surface) the horizontal thrust acting on the wall will be approximately doubled. [14]

If the soil is fully water logged, the submerged unit weight of the whole section of the soil shall be considered. If the soil is partly submerged, the submerged unit weight below the water table will have to be taken into account in both the active and passive states.



(a) (b)

Figure 2.9: Active earth pressure for a backfill; a) partially submerged b) fully submerged

In Fig. 2.9(a), the water table is located at a depth of H_1 below the top of the wall. The unit weight of the backfill is γ in the moist state above the water table and $\gamma^{'}(=\gamma_{sat}-\gamma_w)$ is the effective unit weight below the water table. The unit weight of water is γ_w.

At any section below the top of the wall, up to depth H1 (from Eqn: 2-20),

$\sigma'_{ha} = k_a \gamma z$

Effective vertical pressure, below H1, (i.e. $z > H_1$)

$$\sigma'_v = \gamma H_1 + \gamma'(z - H_1) \quad (2.23)$$

$$\therefore \quad \sigma'_{ha} = K_a \gamma H_1 + K_a \gamma'(z - H_1) \quad (2.24a)$$

Total $$\sigma'_{ha} = K_a \gamma H_1 + K_a \gamma'(z - H_1) + \gamma_w (z - H_1) \quad (2.24b)$$

2.5.1.2. Effect of a surface surcharge load

A uniform surcharge load (q) applied over the whole surface, i.e. very wide and long, and right up to the wall, may be assumed to cause an equal increase in vertical effective stress at all depths:

At depth = z:

Vertical effective stress, $\sigma'_v = \gamma' z + q$ (2.25a)

Active pressure, $\sigma'_{ha} = k_a(\gamma' z + q)$ (2.25b)

Passive pressure, $\sigma'_{hp} = k_p(\gamma' z + q)$ (2.25c)

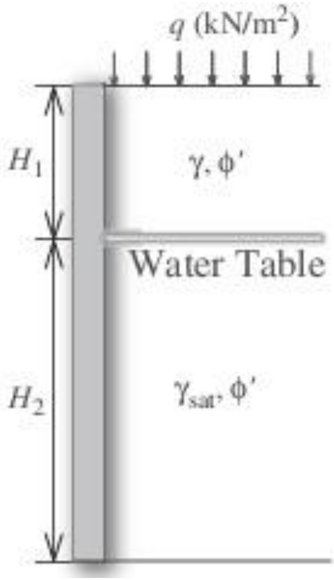
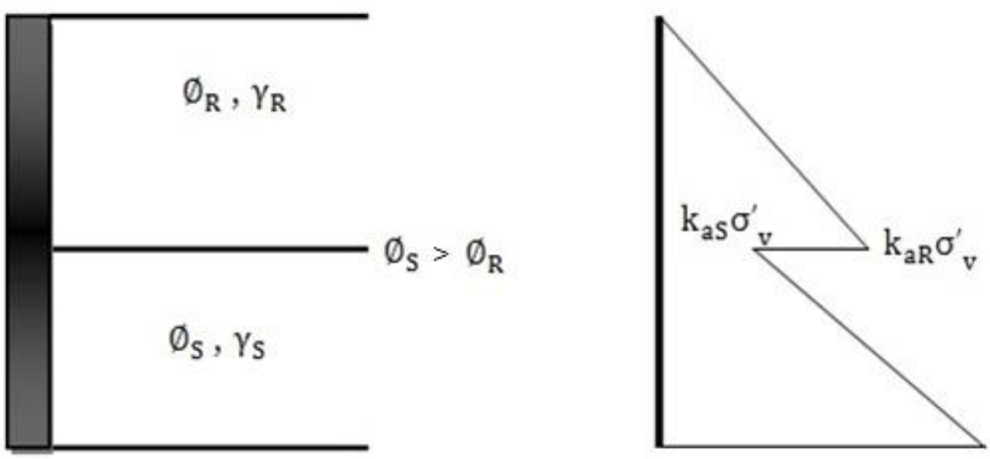


Figure 2.10: Back fill with a surcharge and partially submerged.

For surcharge loading of limited extent, e.g. Point loads, line loads, strip loads and area loads, a more rigorous analysis is required. Such cases shall not be discussed in this study.

2.5.1.3. Effect of stratified soil

Where the soil behind a retaining wall consists of two or more layers, the lateral pressure distribution is determined within each layer and a composite (approximate) diagram drawn (Fig. 2.11b). Suppose at an interface between two layers (R overlying S) the soil properties are $\emptyset_R, \gamma_R$ and $\emptyset_s, \gamma_s$, respectively, and that the vertical effective stress is σ'_v. Then, immediately above the interface $\sigma'_{ha} = k_{aR}\sigma'_v$ and immediately below it $\sigma'_{ha} = k_{aS}\sigma'_v$. The pressure distribution shows a sudden jump in lateral pressure because of the different values of ∅' in the two layers.



(a) (b)

Figure 2.11: Effect of soil stratification on lateral pressure

2.5.1.4. Effect of sloping ground surface

Where the ground surface is sloping, the vertical stress at a given depth will have a value of $\sigma'_v = \gamma z$ for an unloaded surface (Fig. 2.12). The lateral earth pressure against a smooth vertical wall is assumed to act parallel to the ground surface. The relationship between the lateral and vertical stresses can again be obtained analytically using the Mohr circle.

Active pressure, $\sigma'_{ha} = k_a \sigma'_v$

Passive pressure, $\sigma'_{hp} = k_p \sigma'_v$

where: $K_a = \cos\beta \frac{\cos\beta - \sqrt{\cos^2\beta - \cos^2\emptyset'}}{\cos\beta + \sqrt{\cos^2\beta - \cos^2\emptyset'}}$ (2.26)

[14]

$K_p = \cos\beta \frac{\cos\beta + \sqrt{\cos^2\beta - \cos^2\emptyset'}}{\cos\beta - \sqrt{\cos^2\beta - \cos^2\emptyset'}}$ (2.27)

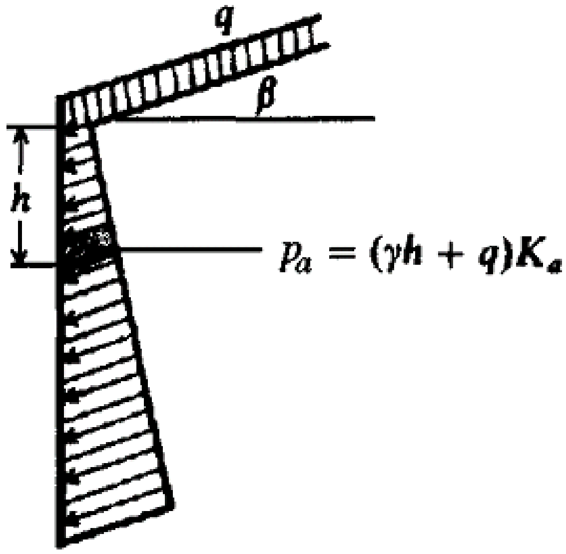


Figure 2.12: Active distribution under sloping ground [**14**]

The horizontal and vertical components of the net active (P_A) and passive (P_p) forces are usually required for design, giving:

$P_{ah} = P_a \cos\beta$ (2.28a)

$P_{av} = P_a \sin\beta$ (2.28b)

$P_{ph} = P_p \cos\beta$ (2.28c)

$P_{pv} = P_p \sin\beta$ (2.28d)

One should not use the Rankine method for K_p when $\beta > 0$, since an inspection of Table 2-1shows that it decreases with increasing β. This is clearly not correct-K_p does properly increase.

Table 2.1: Rankine passive earth pressure coefficients[**14**]

β	ϕ = 26	28	30	32	34	36	38	40	42
0	2.5611	2.7698	3.0000	3.2546	3.5371	3.8518	4.2037	4.5989	5.0447
5	2.5070	2.7145	2.9431	3.1957	3.4757	3.7875	4.1360	4.5272	4.9684
10	2.3463	2.5507	2.7748	3.0216	3.2946	3.5980	3.9365	4.3161	4.7437
15	2.0826	2.2836	2.5017	2.7401	3.0024	3.2926	3.6154	3.9766	4.3827
20	1.7141	1.9176	2.1318	2.3618	2.6116	2.8857	3.1888	3.5262	3.9044
25	1.1736	1.4343	1.6641	1.8942	2.1352	2.3938	2.6758	2.9867	3.3328
30	—	—	0.8660	1.3064	1.5705	1.8269	2.0937	2.3802	2.6940
35	—	—	—	—	—	1.1239	1.4347	1.7177	2.0088
40	—	—	—	—	—	—	—	0.7660	1.2570

2.5.1.5. Effect of Inclined back of wall

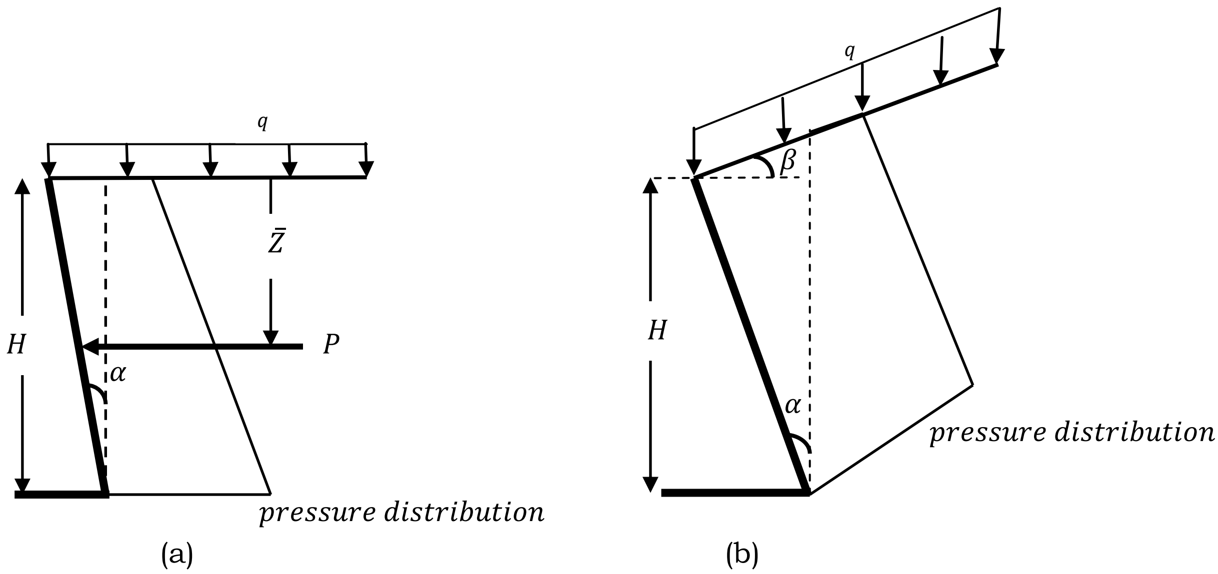


Figure 2.13: Effect of inclined back of wall on lateral earth pressure

The back of a retaining wall may not always be vertical, but may occasionally be battered or inclined. In such case, the lateral earth pressure is computed on an imaginary vertical surface passing through the heel.[16]

This procedure is applicable whether the backfill surface is horizontal or inclined, as illustrated in the above figures.

2.5.2. Lateral pressure in cohesive soils

2.5.2.1. Bell's Theory for cohesive/mixed soils

Rankine's theory deals with earth pressure in cohesionless granular materials, i.e. assuming $c^{'} = 0$. However, in undrained cohesive soils, and in over-consolidated clays where $c^{'} > 0$, the shear strength is expressed wholly or partly in terms of the apparent cohesion:

Undrained shear strength in saturated silts and clays:

$$\tau_f = c_u \ (or \ \tau_f = s_u) \qquad (2.29)$$

Drained shear strength in overconsolidated clays:

$$\tau_f = c^{'} + \sigma'_n \tan\emptyset^{'} \qquad (2.30)$$

In 1915, Bell published an extension of Rankine's theory which included the cohesion parameter.

Drained conditions in cohesive soils:

Horizontal active pressure: $\sigma'_{ha} = K_a\sigma'_v - 2c^{'}\sqrt{K_a}$ (2.31a)

Horizontal passive pressure: $\sigma'_{hp} = K_p\sigma'_v + 2c^{'}\sqrt{K_p}$ (2.31b)

Where $K_a \ and \ K_p$ are as defined in eqns;[$2.19a$] and [$2.19b$].

Undrained conditions in cohesive soils (in terms of TOTAL stresses):

Horizontal active pressure: $\sigma_{ha} = \sigma_v - 2c_u \quad (= \sigma_v - 2s_u)$ (2.32a)

Horizontal passive pressure: $\sigma_{hp} = \sigma_v + 2c_u \quad (= \sigma_v + 2s_u)$ (2.32b)

2.5.2.2. Tension cracks in cohesive soils

The lateral earth pressures under undrained conditions in cohesive soils are:

Horizontal active pressure: $\sigma_{ha} = \sigma_v - 2c_u$

Horizontal passive pressure: $\sigma_{hp} = \sigma_v + 2c_u$

Near the surface, where $\sigma_v < 2c_u$, the active pressure will therefore have a negative or tension value. This of course is an internal tension only and is not transmitted to the supporting wall surface. In fact, if the soil dries slightly tension cracks will open from the surface downward to a depth where the active earth is zero, i.e. $\sigma_v = 2c_u$.

Figure 2.14 shows the distribution of active earth pressure against a smooth vertical wall.

At the ground surface, z = 0 and $\sigma_{ha} = -2c_u$.

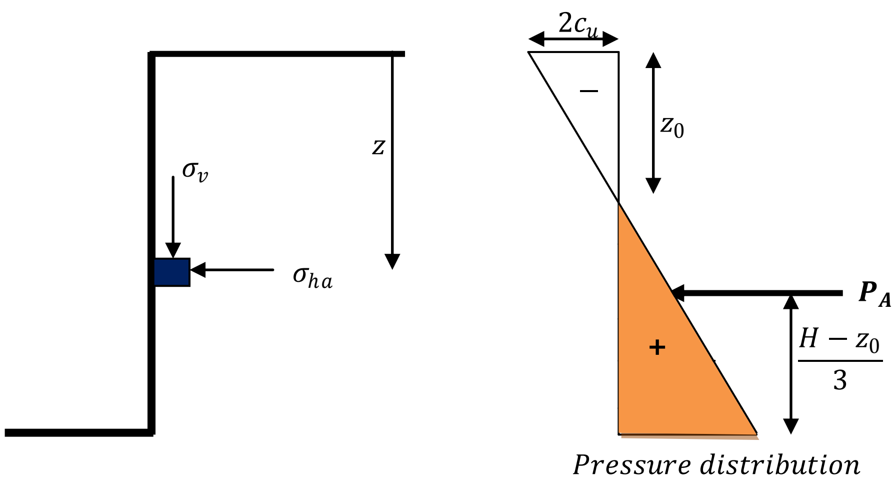


Figure 2.14: Distribution of active pressure in undrained cohesive soil

The tension zone extends down to $= z_o$; here $\sigma_{ha} = 0 \quad = \sigma_v - 2c_u = \gamma z_0 - 2c_u$

Thus the tension crack depth is: $z_o = \frac{2c_u}{\gamma}$ (2.33)

In drained conditions where $\emptyset'$>0, the following expression may be used:

Since $\sigma'_{ha} = 0 = \gamma z_o K_a - 2c\sqrt{K_a}$:

$$z_o = \frac{2c'\sqrt{k_a}}{k_a\gamma} = \frac{2c'}{\gamma\sqrt{k_a}} \quad (2.34)$$

Since the negative earth pressure ('tension') in the soil cannot be transmitted to the supporting wall, it is ignored in the calculation of the active lateral thrust.

The value of P_A is therefore taken as the area of the (positive) shaded portion shown in Fig. 2.14.

Resultant active thrust, $P_A = \frac{1}{2}\sigma'_{ha}(H - z_0)$ (2.35a)

If the surface is unloaded: $P_A = \frac{1}{2}\gamma(H - z_0)^2$ (2.35b)

Where $z_o = \frac{2c_u}{\gamma}$

The line of action of P_A acts through the centre of positive pressure area:

$$\bar{h} = \frac{1}{3}(H - z_o) \quad (2.36)$$

2.5.2.3. Hydrostatic pressure due to water in tension crack

One should not rely on the tension zone (see Fig. 2.14) to reduce lateral pressures. Instead one should assume that it can form and will possibly fill with water. The depth of water (not the quantity) can increase the overturning pressure against the wall considerably owing to both the hydrostatic force of $\frac{1}{2}\gamma_w {z_o}^2$ and the larger moment arm caused by combining the hydrostatic force with the already existing lateral pressure.

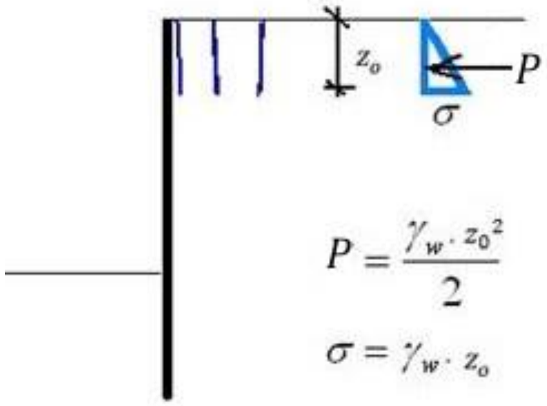


Figure 2.15: Influence of water in tension crack

2.5.2.4. Passive resistance in undrained clay

The distribution of passive pressure in undrained clay is shown in Fig. 2.16. The total passive resistance provided by the soil, with the surface unloaded, is given by area shown below:

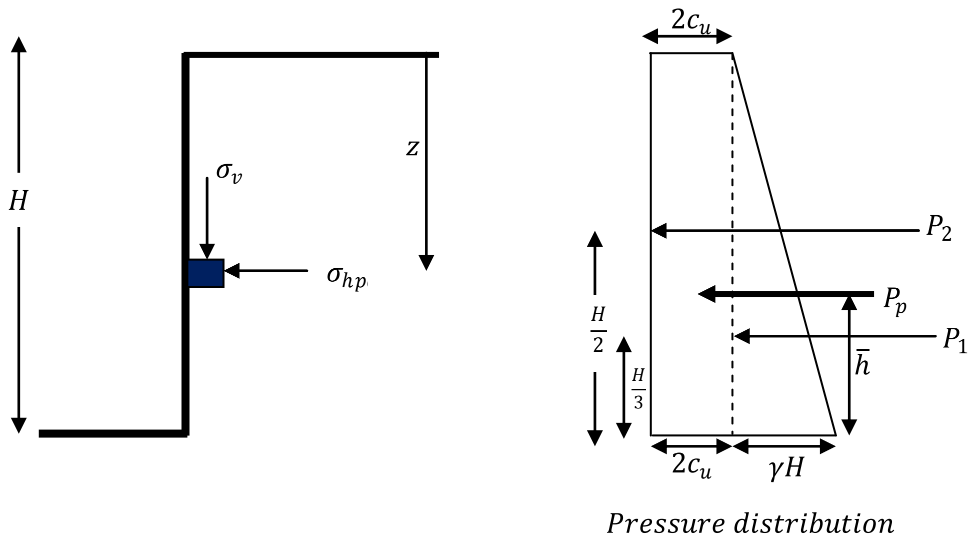


Figure 2.16: Distribution of passive pressure in an undrained cohesive soil

Total passive resistance force $P_p = p_1 + p_2 = \frac{1}{2}\gamma H^2 + 2c_u H$ (2.37)

The line of action of P_p can be obtained by taking moments about the base of the wall:

$$\bar{h} = \frac{P_1 \times \frac{1}{3}H + P_2 \times \frac{1}{2}H}{P_1 + P_2} \quad (2.38)$$

2.5.2.5. Effect of uniform surface surcharge (undrained)

If an extensive uniform surcharge (q) is applied to the surface the vertical and horizontal stresses are:

Vertical total stress, $\sigma_v = \gamma z + q$ (2.39a)

Horizontal active stress, $\sigma_{ha} = \gamma z + q - 2c_u$ (2.39b)

Horizontal passive stress, $\sigma_{ha} = \gamma z + q + 2c_u$ (2.39c)

In the active case, the effect of the surcharge is to reduce the depth of the tension zone. Two forms of active pressure distribution are possible, depending on whether $q < 2c_u$ or q > 2cu (Fig. 2.17). These may be distinguished by evaluating the tension depth z_o: a positive value results when q < 2cu (Fig. 2.17a) and negative value when $q > 2c_u$ (Fig. 2.17b).

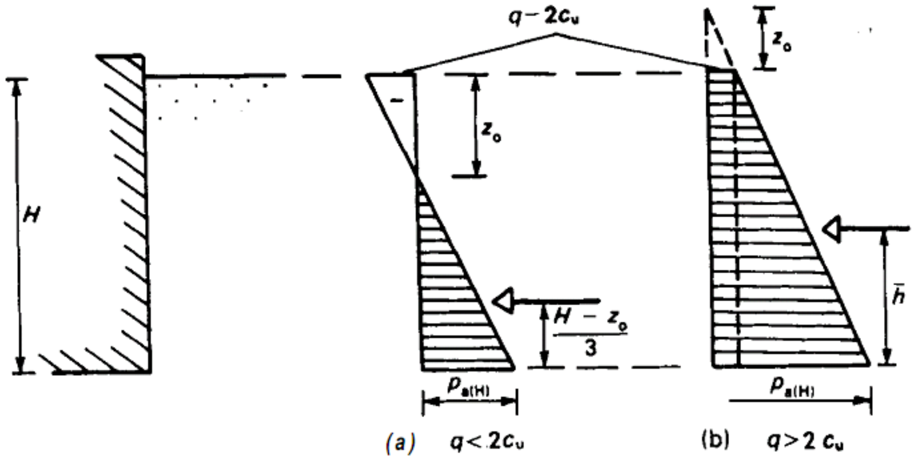


Figure 2.17: Effect of surcharge on undrained active pressure distribution

To obtain z_o, put $\sigma_{ha} = 0$ and $z = z_o$.

Then $\sigma_{ha} = 0 = \gamma z_o + q - 2c_u$

giving; $z_o = \frac{2c_u - q}{\gamma}$ (2.40)

In either case, the resultant thrust is given by the positive area of the diagram.

For $q < 2c_u$: $P_A = \frac{1}{2}\gamma(H - z_o)^2$ (2.41)

For $q > 2c_u$: $P_A = \frac{1}{2}\gamma(H - z_o)^2 - \frac{1}{2}\gamma {z_o}^2$

$= \frac{1}{2}\gamma(H - 2z_o)H$ (2.42)

2.6. Coulomb's Theory and Rough Walls

Rankine’s theory provides a convenient form of analysis that lends itself easily to simple calculations, but it has its limitations. Since neither the nature nor the orientation of the wall is considered, calculations can only be done (strictly) in respect of a smooth vertical wall. The slope of the wall back and the development of friction between the soil and the wall are ignored. Thus, calculated values are somewhat pessimistic. Rankine’s theory is classed as a lower bound solution: the yielding of the whole structure is assumed to coincide with the yielding of the first element.

Some 80 years before Rankine published his theory, Coulomb (1776) had suggested a solution based on a wedge of soil actively moving towards the wall and sliding down a planar slip surface. Thus, the limiting condition is the yielding of the whole wedge: this represents an upper bound solution. Although Coulomb presented a solution only in terms of total stress, he described the shear strength of the soil in terms of both internal friction and cohesion. (It was nearly 140 years later, in 1915, before Bell published his analysis for cohesive soil.)

If a wall supporting a granular soil were not to be there, the soil will slump down to its angle of repose or internal friction. it is therefore reasonable to assume that if the wall only moved forward slightly a rupture plane would develop somewhere between the wall and the surface of repose. The triangular mass of soil between this plane of failure and back of the wall is referred to as the “sliding wedge”. It is reasoned that, if the retaining wall was suddenly removed, the soil within the sliding wedge would slump downward. Therefore, an analysis of forces acting on the sliding wedge at incipient failure will reveal the thrust from the lateral earth pressure which is necessary for the wall to withstand in order to hold the soil mass in place.

2.6.1. Active Pressure on Rough walls

2.6.1.1. Drained condition (C’=0)

Consider the wedge of soil shown in Fig. 2.18a. If the wall were to yield away, bringing the soil to the limiting active state, the wedge ABC will slide down the slip surface AC and towards the wall.

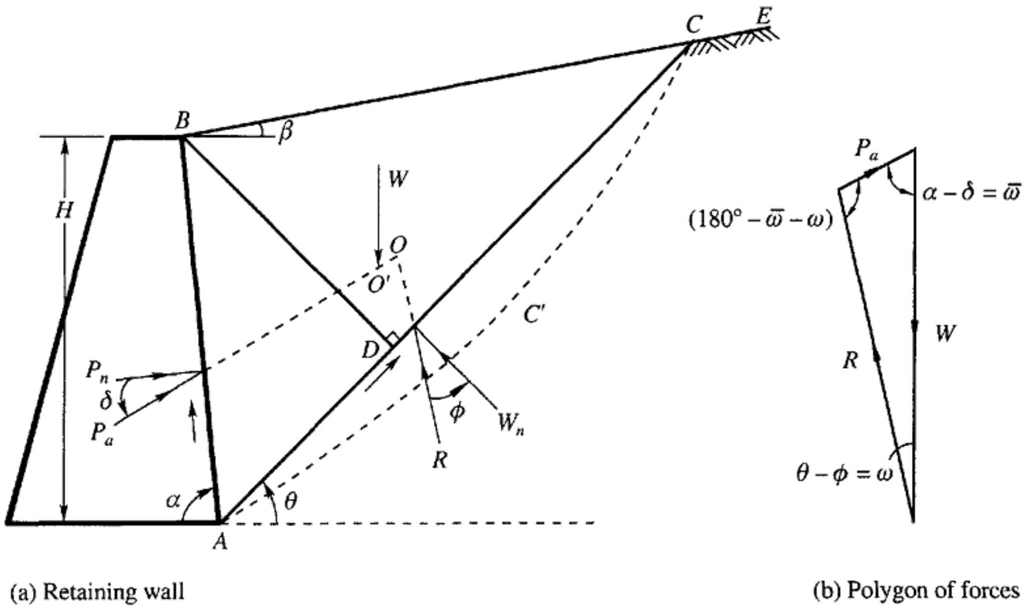


Figure 2.18: Conditions for failure under active conditions

The polygon of forces is shown in Fig. 2.18(b). From the polygon of forces, we may write:

The maximum value of P_a so obtained as:

$$P_a = \frac{1}{2}\gamma H^2 K_a \tag{2.43}$$

where K_a is the active earth pressure coefficient.

$$K_a = \frac{\sin^2(\emptyset+\alpha)}{\sin^2\alpha \sin(\alpha-\delta)\left[1+\sqrt{\frac{\sin(\emptyset+\delta)\sin(\emptyset-\beta)}{\sin(\alpha-\delta)\sin(\alpha+\beta)}}\right]^2} \tag{2.44}$$

The total normal component P_n of the earth pressure on the back of the wall is:

$$P_n = P_a \cos\delta = \frac{1}{2}\gamma H^2 K_a \cos\delta \tag{2.45}$$

If the wall is vertical and smooth, and if the backfill is horizontal, we have:

$$\delta = \beta = 0 \ and \ \alpha = 90$$

Substituting these values in Eq. (2.44), we have:

$$K_a = \frac{1-\sin\emptyset}{1+\sin\emptyset} \tag{2.46}$$

The coefficient K_a in Eq. (2.46) is the same as Rankine's. The effect of wall friction is frequently neglected where active pressures are concerned. Table 2.2 makes this clear. It is clear from this table that K_a decreases with an increase of δ and the maximum decrease is not more than 10 percent.

Table 2.2: Active earth pressure coefficients K_a for $\boldsymbol{\beta = 0 \ and \ \alpha = 90}$ [**17**]

ϕ^0	15	20	25	30	35	40
$\delta = 0$	0.59	0.49	0.41	0.33	0.27	0.22
$\delta = +\phi/2$	0.55	0.45	0.38	0.32	0.26	0.22
$\delta = +/2/3\phi$	0.54	0.44	0.37	0.31	0.26	0.22
$\delta = +\phi$	0.53	0.44	0.37	0.31	0.26	0.22

2.6.1.2. Undrained Conditions ($\emptyset_u$ = 0 and$\tau_f = C_u$)

Undrained conditions can be assumed to occur behind walls supporting saturated clays when the construction period is short and little or no dissipation of excess pore pressure may be expected for some time following. In such cases, short-term stability should be considered in terms of total stresses and the undrained shear strength of the soil, i.e. $\tau_f = c_u$. Consider the case of a vertical rough wall and a horizontal soil surface (Fig. 2.19(a)). A plane failure surface is assumed (BT) which terminates at the bottom of a tension crack of depth z_o. At the limiting active state, the equilibrium of the wedge ABTC is maintained by the following forces.

W = weight of wedge $ABTC = \frac{1}{2}\gamma(H^2 - {z_0}^2)\cot\theta$

R = normal reaction on the failure plane (value not required)

P_A= active thrust acting on the wall

F_c = shear resistance force along the failure plane $BT = c_u(H - z_o)cosec\theta$

F_w= shear resistance force along the wall face = $c_w(H - z_o)$

P_w= horizontal thrust due to water in tension crack = $\frac{1}{2}\gamma_w {z_o}^2$

In which: c_u = undrained shear strength of supported soil
c_w = undrained adhesion between soil and wall

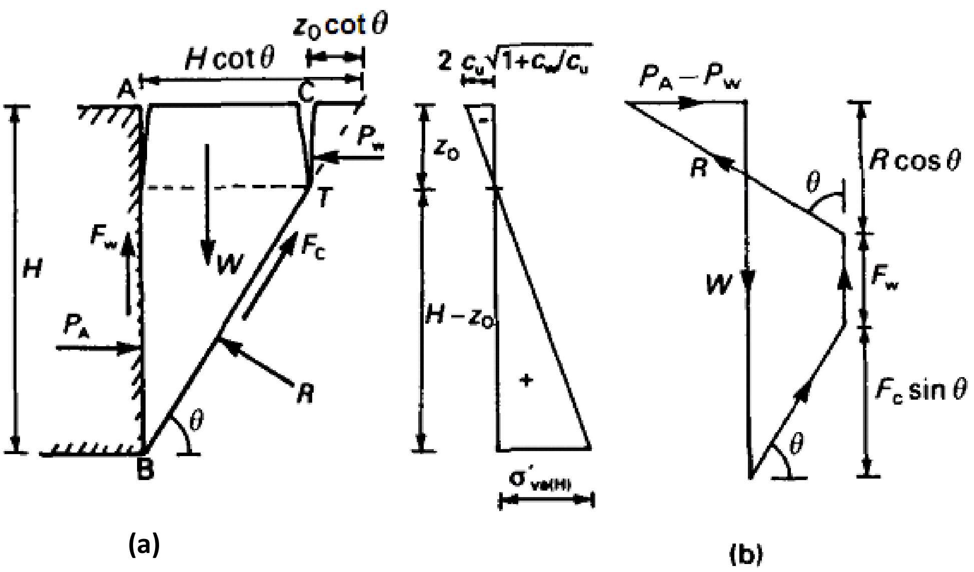


Figure 2.19: Coulomb's theory for undrained conditions

Wall adhesion [14]

The magnitude of wall adhesion (c_w) varies between $0.3c_u$ for stiff clays to c_u for soft clays, but the extent of its development along the wall surface is problematical. In the absence of empirical evidence, it is usual to adopt a value of $0.5c_u$.

The polygon of forces acting on the wedge is shown in Fig.2.19(b), from which:

$$P_A = \frac{1}{2}\gamma(H^2 - {z_o}^2) - 2c_u(H - z_o)\sqrt{1 + \frac{c_w}{c_u}} + P_w \qquad (2.47)$$

or putting $K_{ac} = 2\sqrt{1 + \frac{c_w}{c_u}}$

$$P_A = \frac{1}{2}\gamma(H^2 - {z_0}^2) - c_u(H - z_o)K_{ac} + P_w \qquad (2.48)$$

At a given depth the lateral pressure is :

$$\sigma'_{ha} = \gamma z - c_u K_{ac} = \gamma z - 2c_u\sqrt{1 + \frac{c_w}{c_u}} \qquad (2.49)$$

At the bottom of the tension zone, $z = z_o \ and\ \sigma'_{ha} = 0$

Therefore, $z_o = \frac{2c_u\sqrt{1+\frac{c_w}{c_u}}}{\gamma}$ (2.50)

or with a uniform surcharge q:

$$z_o = \frac{2c_u\sqrt{1+\frac{c_w}{c_u}} - q}{\gamma} \qquad (2.51)$$

2.6.1.3. Drained Overconsolidated Clay (C' > 0)

In examining the stability of a mass of heavily overconsolidated clay it is necessary to consider the potential amount of strain associated with the onset of failure. The peak strength ($\tau_f = c' + \sigma'_n \tan \emptyset'_f$) of the soil should be invoked only when it is known that very small strains precede failure. In the majority of cases involving retaining walls, the pre-yield strains are likely to be high enough to produce significant expansion. Thus, failure will occur at the critical state $\tau_f = \sigma'_n \tan \emptyset'\ i.e.\, c' = 0$.

As a general rule, therefore, the active thrust should be calculated using ;

$$P_A = \frac{1}{2}\gamma H^2 K_a - c' H K_{ac} \qquad (2.52)$$

Where; K_a is earth pressure coefficients as before and;

$$K_{ac} = \sqrt{k_a(1+\frac{c_w}{c})} \quad (2.53)$$

Where; c_w = wall adhesion

c' = effecive cohesion

At a given depth the lateral pressure is :

$$\sigma'_{ha} = \gamma z K_a - c'K_{ac} = \gamma z K_a - 2c'\sqrt{k_a\left(1+\frac{c_w}{c'}\right)} \quad (2.54)$$

At the bottom of the tension zone, $z = z_o$ and $\sigma'_{ha} = 0$

Therefore, depth of a dry tension crack is given by:

$$z_o = \frac{2c\sqrt{1+\frac{c_w}{c}}}{\gamma\sqrt{K_a}} \quad (2.55)$$

2.6.2. Passive pressure on rough walls

2.6.2.1. Drained conditions (C' = 0)

Coulomb's theory may be used to establish passive pressures and thrust on a rough wall that has an inclined back and where the ground surface is a regular inclined plane. The analysis follows along the same lines as that for active pressures described above. The following expressions are obtained:

$$P_p = \frac{1}{2}\gamma H^2 K_p \quad (2.56)$$

Where:

$$K_p = \frac{\sin^2(\emptyset+\alpha)}{\sin^2\alpha\sin(\alpha+\delta)\left[1+\sqrt{\frac{\sin(\emptyset+\delta)\sin(\emptyset+\beta)}{\sin(\alpha+\delta)\sin(\alpha+\beta)}}\right]^2} \quad (2.57)$$

Thrust component normal to the wall is given by:

$$P_n = P_p\cos\delta = \frac{1}{2}\gamma H^2 K_p\cos\delta \quad (2.58)$$

Definitions of angles, etc. are given in Fig. 2.20. For problems where the soil and wall surfaces are simple planes, the point of application of P_p, may be taken as occurring at H/3 above the base. For a smooth vertical wall and a horizontal unsurcharged ground surface, i.e. $\alpha = 90°, \beta = 0$ and $\delta = 0$ the solution obtained for K_p will be the same as that given by the Rankine expression, i.e. $\frac{1+\sin \emptyset}{1-\sin \emptyset}$.

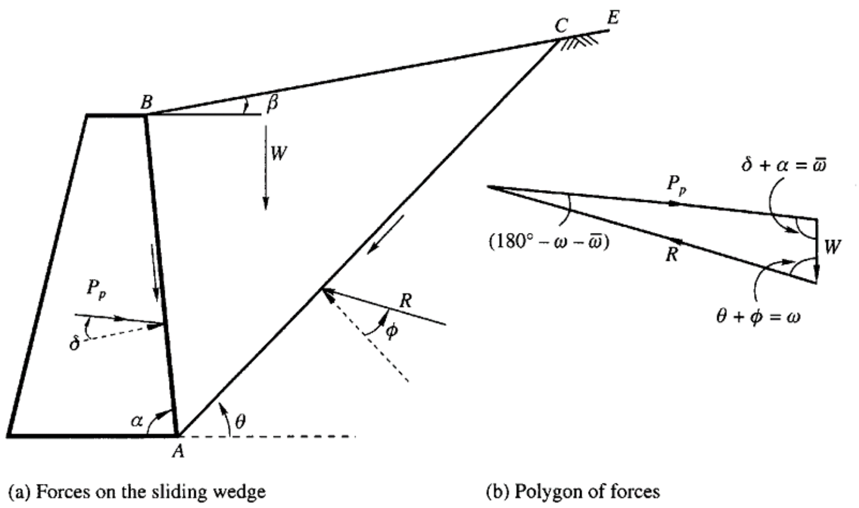


Figure 2.20: Conditions for failure under passive conditions

Limitations of the Coulomb theory[5][7][14]

When the value of δ is greater than about $1/3\,\emptyset'$, the failure surface is significantly curved at the lower end near the base of the wall. The use of Coulomb's theory in such cases, since it is based on a planar failure surface, leads to a marked overestimation of the passive resistance. For the active case the error in assuming a plane surface is small and K_a is under-estimated slightly.

A number of alternative methods have been proposed wherein the curved part is assumed variously to be a circular arc, an ellipse or a logarithmic spiral. One of the methods is discussed in the sub- section to come.

2.6.2.2. Undrained Conditions ($\emptyset_u$ = 0)

As in the drained analysis, the effect of frictional resistance tangential to the wall face produces curvature in the failure surface near the base of the wall. However, the errors incurred by assuming a planar failure surface in this (undrained) case are less significant. Figure 2.21 shows the arrangement of forces.

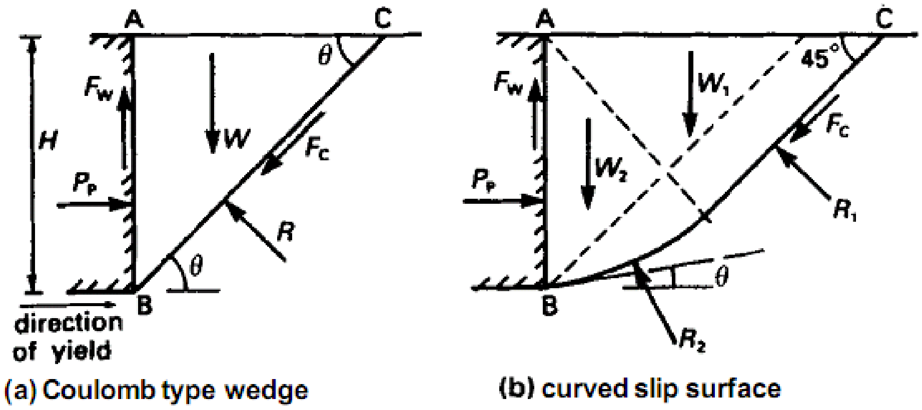


Figure 2.21: Passive pressure in undarained conditions

The analysis here is similar to that given in sub-section 2.6.1.2 except that no tension crack will occur.

Thus,

$$P_{pu} = \frac{1}{2}\gamma H^2 + c_u H K_{pc} \tag{2.59}$$

$$\sigma'_{hp} = \gamma z + c_u K_{pc} = \gamma z + 2c_u\sqrt{\left(1+\frac{c_w}{c_u}\right)} \tag{2.60}$$

2.6.2.3. Drained Overconsolidated Clay (c' > 0)

Provided that the strains preceding the onset of failure are considered to be low (< 2 percent), the peak strength ($\tau_f = c' + \sigma'_n \tan \emptyset'$) may be invoked as the shear stress at the moment of yielding. In cases where the pre-yield strains are likely to be significantly large, the critical state strength ($\tau_f = \sigma'_n \tan \emptyset'$) should be used, i.e. putting c' = 0.

The passive thrust at low lateral strain is given by;

$$P_p = \frac{1}{2}\gamma H^2 K_p + c' H K_{pc} \tag{2.61}$$

Where K_p is the earth pressure coefficients as previously described and

$$K_{pc} = 2\sqrt{K_p\left(1+\frac{c_w}{c'}\right)} \tag{2.62}$$

2.6.3. Curved Failure Surface

In the previous section, Coulomb's earth pressure theory, the retaining wall was considered to be rough. The potential failure surfaces in the backfill were considered to be planes. In reality most failure surfaces in soil are curved. There are several instances where assumptions of plane surfaces in soil may provide unsafe results. Examples of these cases are the estimation of passive pressure in braced cuts. Here the works of Kerisel and Absi shall be considered in order to incorporate curved surfaces of failure plane.

2.6.3.1. Kerisel and Absi's theory (1990) [7][13]

To take account of the effects of wall friction Kerisel and Absi have assumed the curved failure surface to be a logarithmic spiral. They have determined the earth pressure coefficients K_a and K_p for a horizontal backfill and vertical wall.

$$p'_{ah} = K_a \sigma'_v - K_{ac} c' \qquad (2.63)$$

$$p'_{ph} = K_p \sigma'_v + K_{pc} c' \qquad (2.64)$$

The above equations are used to compute the horizontal component of pressure p_{ah} and p_{ph}. For the active condition:

$$p_{ah} = p_a \cos \delta \qquad (2.65)$$

Where p_a is the resultant value of pressure and force acting at an angle δ to the normal to back of the wall. The shear force acting on the wall is given by:

$$p_{ah} \tan \delta \qquad (2.66)$$

Kerisel and Absi have produced a chart to give the horizontal components of active and passive pressures.

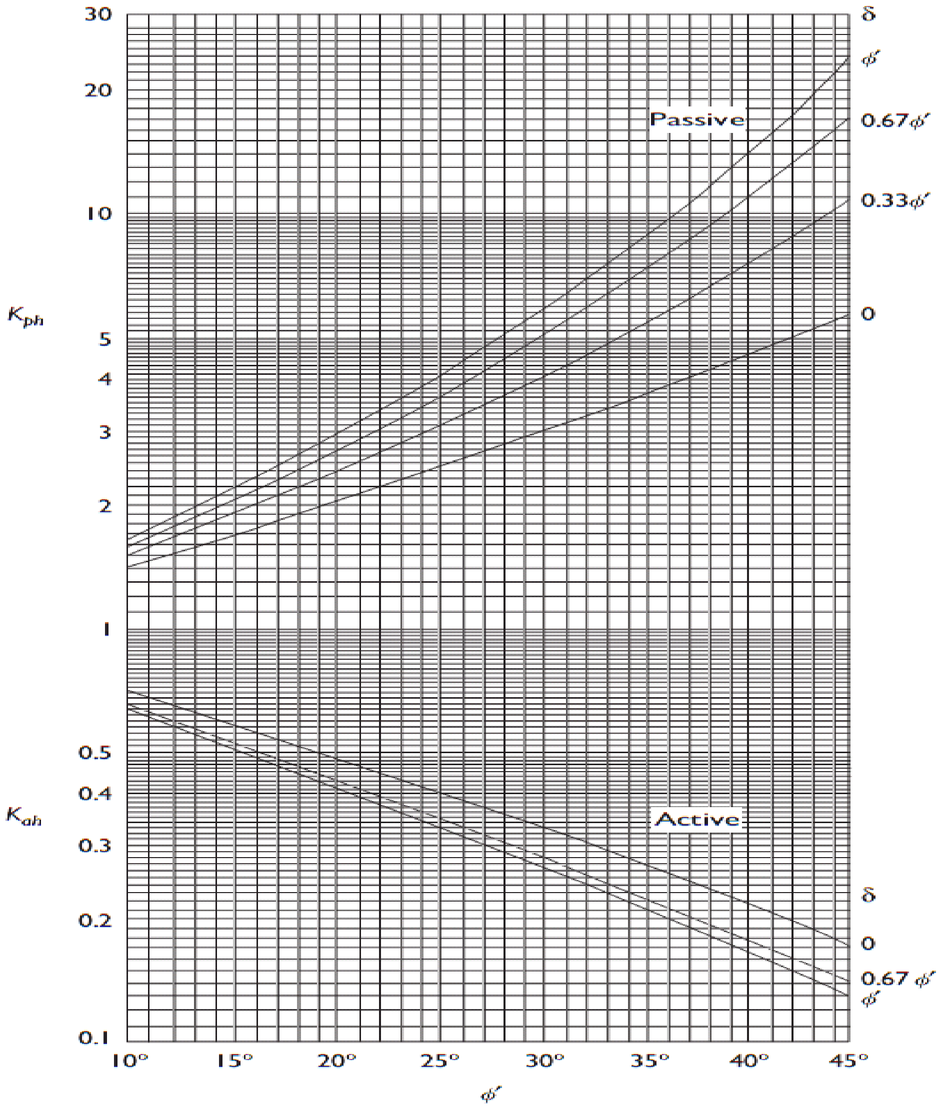


Figure 2.22: Coefficients for horizontal components of active and passive pressure

Values of K_{ac} and K_{pc} can be obtained with sufficient accuracy from the expressions:

$$K_{ac} = 2\sqrt{K_a\left(1 + {c'_w}/{c'}\right)} \quad (2.67)$$

$$K_{pc} = 2\sqrt{K_p\left(1 + {c'_w}/{c'}\right)} \quad (2.68)$$

Where c'_w is the drained or effective stress wall adhesion.

The above expression will be appropriate for granular soils and overconsolidated clays where the critical condition will be the drained case and effective stress parameters are applicable.

The angle of wall friction δ will depend on the frictional characterstics of the soil and roughness of the wall is usually given as a proportion of $\emptyset'$, the value δ never exceeding $\emptyset'$. Commonly adopted values of angle of wall friction and wall adhesion are given by Williams and Waite, 1993.

Table 2.3: Values of skin friction and wall adhesion (from Williams and Waite, 1993)[7]

Analysis	Angle of wall friction		Wall adhesion	
	Active	**Passive**	**Active**	**passive**
Effective stress	$0.67\emptyset'$	$0.5\emptyset'$	$0.5c'$	$0.5c'$
Total Stress	-	-	$0.5c_u$	$0.5c_u$
			$50KN/m^2$	$25KN/m^2$
			Maximum	

2.7. Design of Retaining Walls

Retaining walls are used to prevent retained material from assuming its natural slope. Wall structures are commonly used to support earth, coal, ore piles, and water.

Retaining walls may be classified according to how they produce stability:

1. Mechanically reinforced earth
2. Gravity—either reinforced earth, masonry, or concrete
3. Cantilever—concrete or sheet-pile
4. Anchored—sheet-pile and certain configurations of reinforced earth

At present, the mechanically stabilized earth and gravity walls are probably the most used—particularly for roadwork where deep cuts or hillside road locations require retaining walls to hold the earth in place. These walls eliminate the need for using natural slopes and result in savings in both right-of-way costs and fill requirements.

Cantilever walls of reinforced concrete are still fairly common in urban areas because they are less susceptible to vandalism and often do not require select backfill. Typically they compete well in costs where the wall is short (20 to 50 m in length) and not very high (say, under 4 m). They are also widely used for basement walls and the like in buildings.[10]

In this thesis two of the major retaining wall types, gravity and cantilever, shall be dealt with for their external stability.

2.7.1. Common Proportions of Retaining Walls

The usual practice in the design of retaining walls is to assign tentative dimensions and then check for over all stability of the structure. In this study, software is developed to simplify and quicken the process.

2.7.1.1. Cantilever Wall

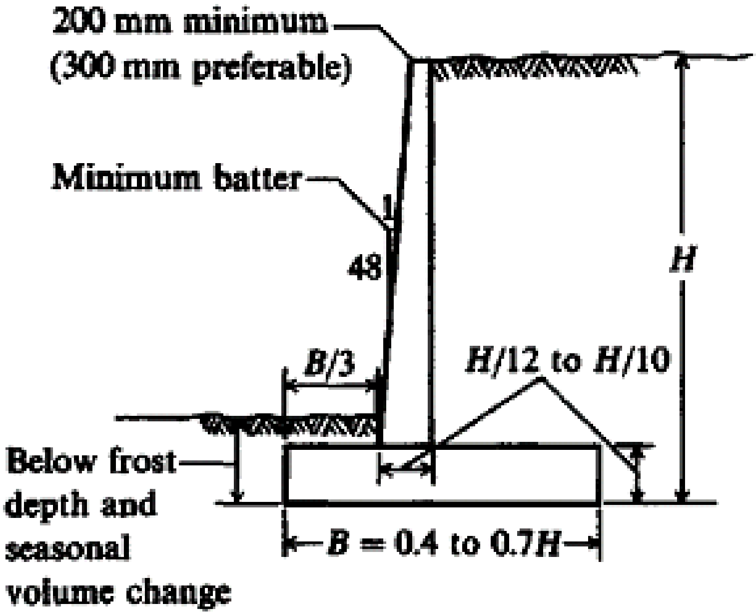


Figure 2.23: Tentative design dimensions for a cantilever retaining wall. (Batter shown is optional)[**10**]

Dimensions for retaining wall should be adequate for structural stability and to satisfy local building-code requirements. The tentative dimensions shown in Fig 2.23 are based heavily on experience accumulated with stable walls and may be used in the absence of other data, but may result in an overly conservative design.

The top of the stem should generally be not less than 200mm, and preferably not less than 300mm, so that proper placement of concrete may be effected and, if part of the end spalls off or is broken off, a sufficient amount will remain to satisfy structural and aesthetic requirement. The base of the stem should be thick enough to satisfy the shear requirements without the use of shear reinforcing steel.

The base-slab dimensions should be such that the resultant of the vertical loads falls within the middle one-third. If the resultant falls outside the middle one-third, the toe pressures will be excessively large. A batter is usually used for cantilever retaining walls to effect some savings in material. A front batter is preferable so that wall movement to develop active pressure is not noticeable. Most highway retaining walls and nearly all bridge abutments have the exposed face vertical. A slight increase in wall stability is usually obtained when the batter is on the back face.

2.7.1.2. Gravity Walls

Gravity-wall dimensions may be taken as shown in figure 2.24. Gravity walls are generally trapezoidal-shaped. The base and other dimensions should be such that the resultant falls within the middle one-third of the base. The top width of the stem should be 20 to 25cm, with 25cm minimum preferred.

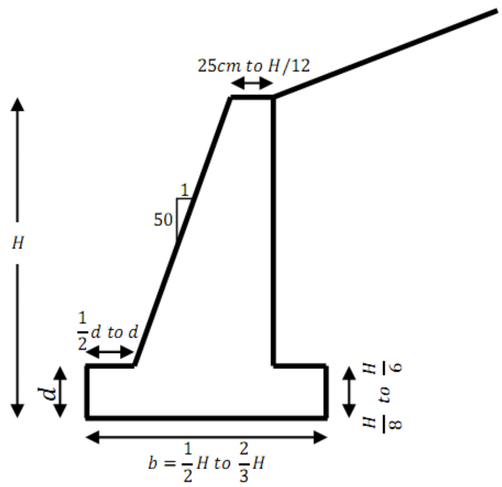


Figure 2.24: Tentative dimensions for gravity retaining wall [**1**][**10**]

2.7.2. Stability of Walls

Retaining walls must provide adequate stability against sliding. The soil in front of the wall provides a passive-earth-pressure resistance as the wall tends to slide into it. If the soil is excavated or eroded after the wall has been built, the passive pressure component is not available and sliding instability may occur. If there is certainty of no loss of the toe soil the designer may use the passive pressure in this zone as part of the sliding resistance.

The sliding resistance along the base is taken as;

$F_r = f * V + c'B + P_{ph}$ (2.69)

Where V includes all the vertical forces, including the vertical component of P_a, acting on the base.

The coefficient of friction between the base and the soil may be taken as:

$$f = \tan\emptyset' \quad to \quad \tan\delta$$

And the base-to-soil adhesion is usually a fraction of the cohesion-values of 0.6 to 0.8 are commonly used.[9]

$$c_b = 0.6c' \quad to \quad 0.8c'$$

The base soil is usually compacted prior to pouring the base slab. The wet concrete will always attach to the ground such that $f = \tan\emptyset'$ is not unrealistic. The cohesion may be considerably destroyed; thus values of 0.6 to 0.8c' are more appropriate.

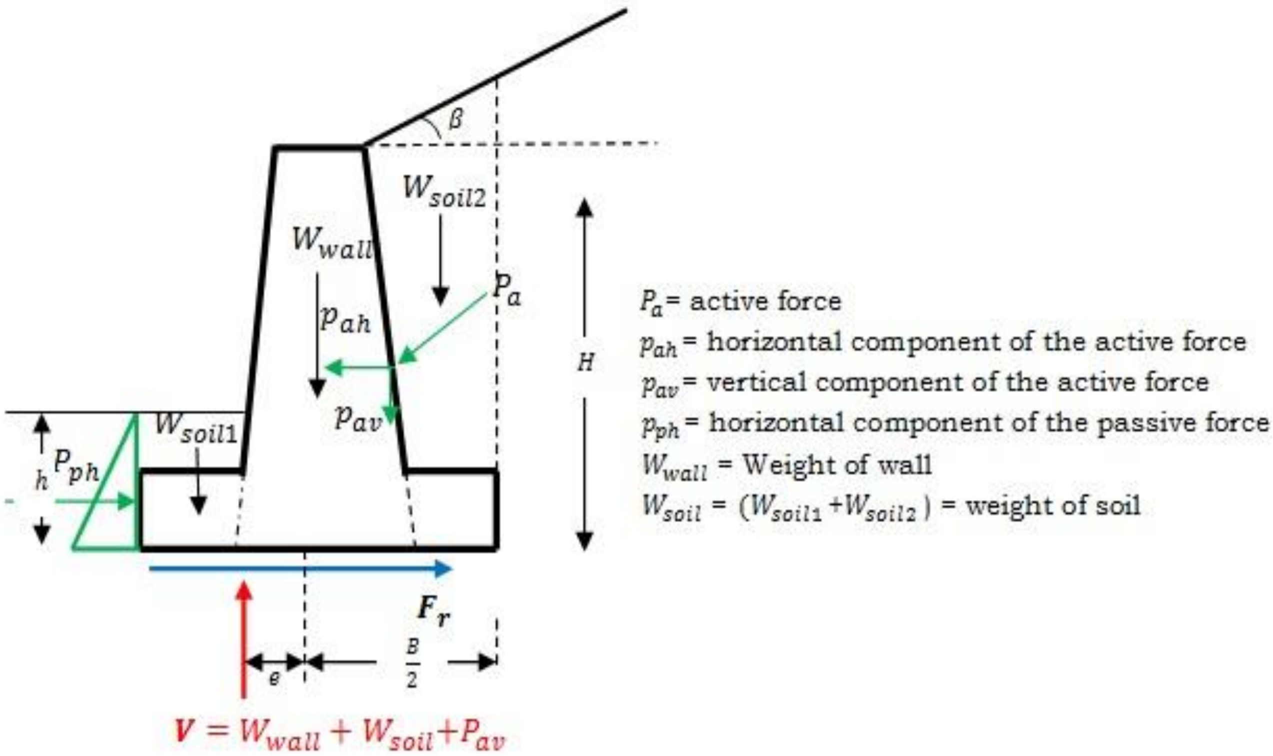


Figure 2.25: Forces involved in the sliding stability of retaining wall [**9**][**10**]

The wall must be safe against sliding. That is, sufficient friction F_r must be developed between the base slab and the base soil that a safety factor $\boldsymbol{FS_{sl}}$ is:

$$FS_{sl} \leq \frac{F_r}{P_{ah}} \tag{2.70}$$

Where, P_{ah}= horizontal component of the active earth pressure at the back of the wall

Taking moments about the toe of the retaining wall, we can compute a safety factor against overturning as:

$$FS_{ot} \leq \frac{sum\ of\ resisting\ moments}{sum\ of\ overturning\ moments} \tag{2.71}$$

2.7.3. Retaining Wall Forces

The forces acting on a retaining wall are customarily taken per unit of width for both gravity and cantilever walls.

2.7.3.1. Gravity Walls

The forces on a gravity wall are as indicated in Fig. 2.26. The active earth pressure is computed either by Rankine or Coulomb methods, as presented in the previous sections. If the coulomb method is used, it is assumed that there is incipient sliding on the back face of the wall, and the earth pressure acts at the angle of wall friction δ to the normal with the wall. The Rankine solution applies to P_a acting at the angle β on a vertical plane through the heel.

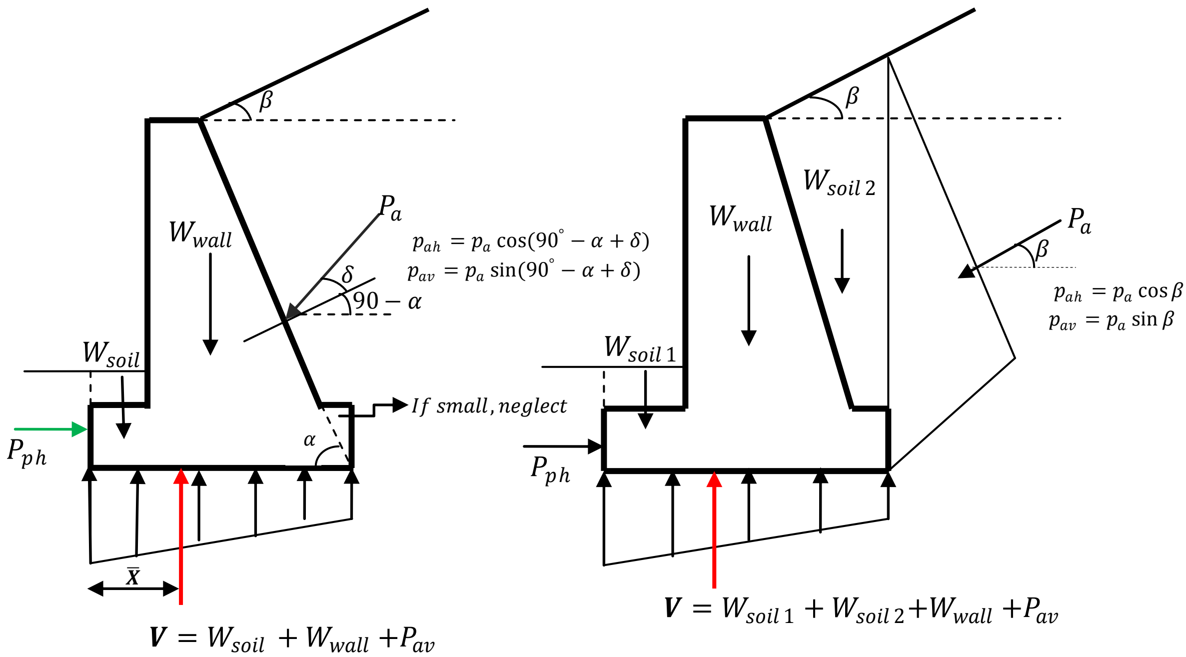


Figure 2.26: Forces on a gravity wall. (a) Coulomb analysis; (b) Rankine analysis [**8**]

The vertical resultant V acting on the base is equal to the sum of the forces acting down ward, and will have an eccentricity e with respect to the geometrical center of the base. Taking moments about the toe,

$$\bar{x} = \frac{sum\ of\ resisting\ \ net\ moments}{V} \quad (2.72)$$

if the width of the base is B, the eccentricity of the base can be computed as;

$$e = \frac{B}{2} - \bar{x} \quad (2.73)$$

2.7.3.2. Cantilever Walls

Forces on a cantilever wall are shown in Fig. 2.27. Note that for computations the total vertical force V is;

$$V = W_{wall} + W_{soil} + P_{a\sin\beta} \quad (2.74)$$

The heel force $P_{av} = P_{a\sin\beta}$ is sometimes not included for a more conservative factor of safety. Use of a passive force P_p could be applicable if the base soil is in close contact with the face of the toe.

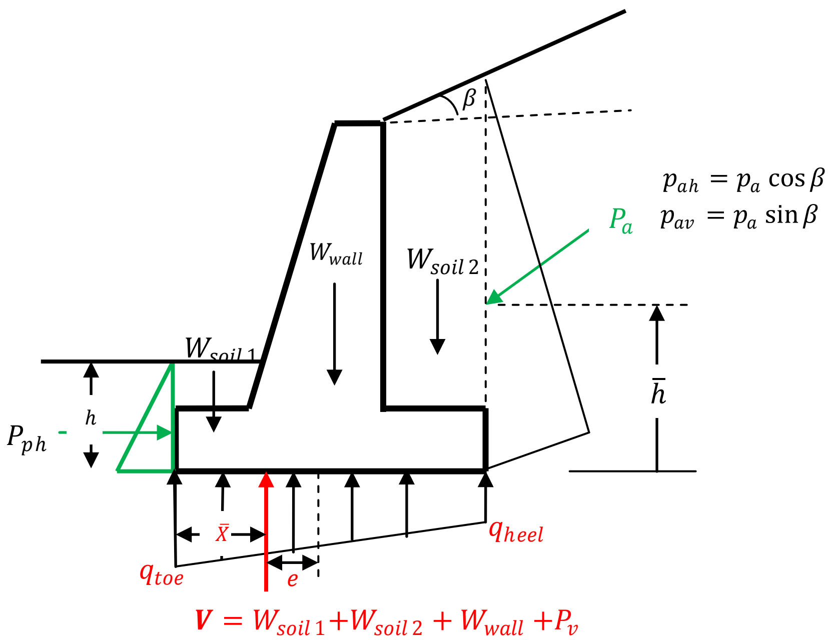


Figure 2. 27: Forces on cantilever wall

2.7.4. BEARING CAPACITY

The foundation of the wall must be designed to avoid bearing-capacity failure. The distribution of ground bearing pressure beneath the base is taken to be either trapezoidal or triangular, the maximum pressure usually occurs beneath the toe. This maximum value must not exceed the allowable bearing capacity of the soil. Hence, to achieve stability against bearing capacity failure;

$$q_{all} \geq q_{max} \tag{2.75}$$

2.7.4.1. The General Bearing Capacity Equation

Meyerhof (1963) suggested the following form of the general bearing capacity equation for the computation of ultimate bearing capacity:[3]

$$q_u = c' N_c F_{cs} F_{cd} F_{ci} + q N_q F_{qs} F_{qd} F_{qi} + 0.5 \gamma B N_\gamma F_{\gamma s} F_{\gamma d} F_{\gamma i} \tag{2.76}$$

In this equation:

c' = cohesion
q = effective stress at the level of the bottom of the foundation
γ = unit weight of soil
B = width of foundation
$F_{cs}, F_{qs}, F_{\gamma s}$ = shape factors
$F_{cd}, F_{qd}, F_{\gamma d}$ = depth factors
$F_{ci}, F_{qi}, F_{\gamma i}$ = load inclination factors
N_c, N_q, N_γ = bearing capacity factors

Bearing Capacity Factors

$$N_q = \tan^2\left(45 + \frac{\emptyset'}{2}\right) e^{\pi \tan \emptyset'} \tag{2.77}$$

$$N_c = (N_q - 1)\cot\emptyset' \quad \text{, When } \emptyset' = 0, N_c = 5.14 \tag{2.78}$$

$$N_\gamma = 2(N_q + 1)\tan\emptyset' \tag{2.79}$$

Shape, Depth, Inclination Factors [3]

Commonly used shape, depth, and inclination factors are given in Table 2.4.

Table: 2. 4 : Shape, depth and Inclination factors

Factor	Relationship	Reference
Shape	$F_{cs} = 1 + \left(\frac{B}{L}\right)\left(\frac{N_q}{N_c}\right)$ $F_{qs} = 1 + \left(\frac{B}{L}\right)\tan\phi'$ $F_{\gamma s} = 1 - 0.4\left(\frac{B}{L}\right)$ For strip foundations, the B/L ratio is set to zero.	DeBeer (1970)
Depth	$\frac{D_f}{B} \leq 1$ For $\phi = 0$: $F_{cd} = 1 + 0.4\left(\frac{D_f}{B}\right)$ $F_{qd} = 1$ $F_{\gamma d} = 1$ For $\phi' > 0$: $F_{cd} = F_{qd} - \frac{1 - F_{qd}}{N_c \tan\phi'}$ $F_{qd} = 1 + 2\tan\phi'(1 - \sin\phi')^2\left(\frac{D_f}{B}\right)$ $F_{\gamma d} = 1$ $\frac{D_f}{B} > 1$	Hansen (1970)

Factor	**Relationship**	**Reference**
	For $\phi = 0$: $F_{cd} = 1 + 0.4 \underbrace{\tan^{-1}\left(\frac{D_f}{B}\right)}_{\text{radians}}$ $F_{qd} = 1$ $F_{\gamma d} = 1$ For $\phi' > 0$: $F_{cd} = F_{qd} - \frac{1 - F_{qd}}{N_c \tan \phi'}$ $F_{qd} = 1 + 2 \tan \phi'(1 - \sin \phi')^2 \underbrace{\tan^{-1}\left(\frac{D_f}{B}\right)}_{\text{radians}}$ $F_{\gamma d} = 1$	
Inclination	$F_{ci} = F_{qi} = \left(1 - \frac{\beta°}{90°}\right)^2$ $F_{\gamma i} = \left(1 - \frac{\beta}{\phi'}\right)$ $\beta =$ inclination of the load on the foundation with respect to the vertical	Meyerhof (1963), Hanna and Meyerhof (1981)

2.7.4.2. Factor of Safety

Calculating the allowable load-bearing capacity of shallow foundations requires the application of a factor of safety (FS) to the ultimate bearing capacity, or;

$$q_{all} = \frac{q_u}{FS} \tag{2.80}$$

2.7.4.3. Modification of Bearing Capacity Equations for Water Table

Equation (2.76) give the ultimate bearing capacity, based on the assumption that the water table is located well below the foundation. However, if the water table is close to the foundation, some modifications of the bearing capacity equations will be necessary. (See Figure 2.28.)[3]

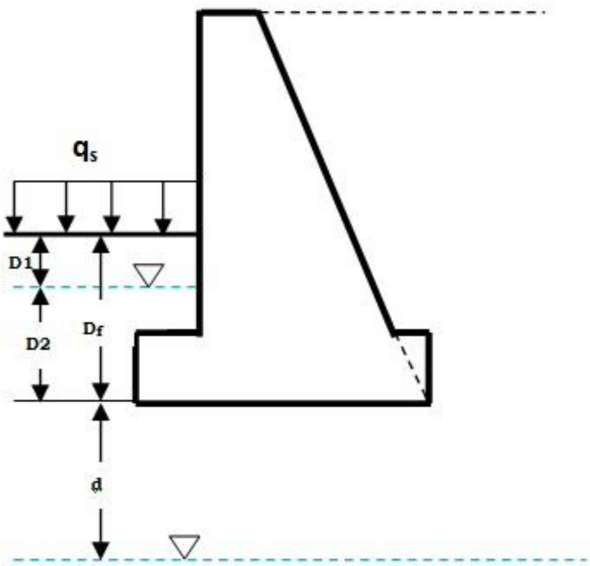


Figure 2.28: Modification of the bearing capacity equation for water table

Case I. If the water table is located so that $0 < D_1 < D_f$ the factor q in the bearing capacity equation takes the form

$$q = effective\ surcharge = q_s + D_1\gamma_{ff} + D_2(\gamma_{ff.sat} - \gamma_w) \quad (2.81)$$

Where;

γ_{ff} = Unit weight of front face soil

$\gamma_{ff.sat}$ = Saturated unit weight of front face soil

γ_w = Unit weight of water

Also, the value of γ in the last term of the general bearing capacity equation has to be replaced by;

$\gamma'_f = \gamma_{fsat} - \gamma_w$, Where $\gamma'_f = effective\ unit\ weight\ of\ the\ foundation\ soil$

$\gamma_{f.sat} = saturated\ unit\ weight\ of\ the\ foundation\ soil$

Case II. For a water table located so that $0 \leq d \leq B$,

$$q = q_s + \gamma D_f \tag{2.82}$$

In this case, the factor in the last term of the bearing capacity equations must be replaced by the factor;

$$\bar{\gamma} = \gamma'_f + \frac{d}{B}(\gamma_f - \gamma'_f) \tag{2.83}$$

Case III. When the water table is located so that $d \geq B$ the water will have no effect on the ultimate bearing capacity.

3. METHODOLOGY

3.1. General

As mentioned earlier, the aim of this thesis is to develop software for analysis of Lateral Earth Pressure and Design of Retaining Walls for external stability. The software is developed using visual basic.net 2010 programming language. Flowchart, which is annexed in this thesis, is first prepared before going to designing the interface of the software and writing the code. The next work performed is preparing interface, writing the code and debugging.

The software has a multi-form user interface. These forms communicate to each other through globally declared variables in a module. Debugging is done repeatedly during the coding time.

The software performs lateral earth pressure analysis for different ground, soil and wall inclination conditions. It also designs gravity and cantilever retaining walls for external stability purposes.

The different scenarios addressed by the software together with series of equations used to solve them are presented as Annex – A.

3.2. Design of Retaining walls

3.2.1. Proportioning

In the previous chapter tentative dimensions, which have been proven to produce a stable retaining wall, have been given. In the GeoLEPARD-software the minimum values of the ranges have been used to be the default dimensions of the retaining walls. The user will have access to edit these default dimensions and check the stability of the walls.

3.2.2. Lateral Earth Pressure Computation

3.2.2.1. Gravity Retaining wall

I. Rakine's (Bell's) and Kerisel and Absi's Theory

While using these theories for the design of retaining walls, the lateral pressure is computed on a vertical plane passing through the heel**.** [**9**]

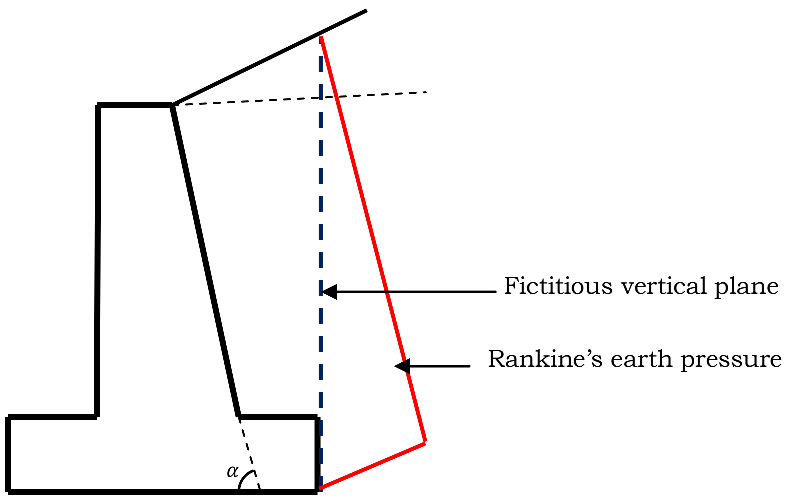


Figure 3.1: Rankine's earth pressure for the design of retaining walls

II. Coulomb's Theory

Coulomb's theory applies to both vertical walls and walls having an inclined back face. And lateral pressure computation can be taken along the back face of the wall. However, the presence of heel in the wall system interferes with failure wedge. In order to deal with such conditions, the following two approaches are adopted[9].

a. If the heel is 15 cm or smaller and any of the soil layers do not involve cohesion, then the heel section is assumed not to exist. And lateral pressure is computed along the back face of the wall.

b. If the heel is more than 15cm, then a fictitious vertical plane is drawn through the heel and lateral pressure is computed about this plane.

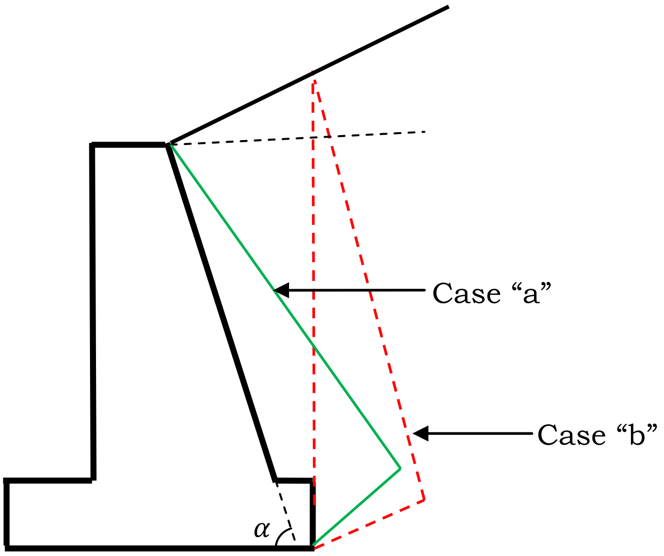


Figure 3.2: Coulomb’s earth pressure theory for the design of retaining walls

3.2.2.2. Cantilever Retaining wall

As cantilever retaining walls have longer heels, they definitely interfere with the failure wedge. Hence, computation of earth pressure, for all the theories discussed, shall be executed about a vertical plane passing through the heel.

3.2.3. Stability Analysis

In this section, wall shall be checked for stability against;

a) Overturning
b) Sliding
c) Bearing capacity failure

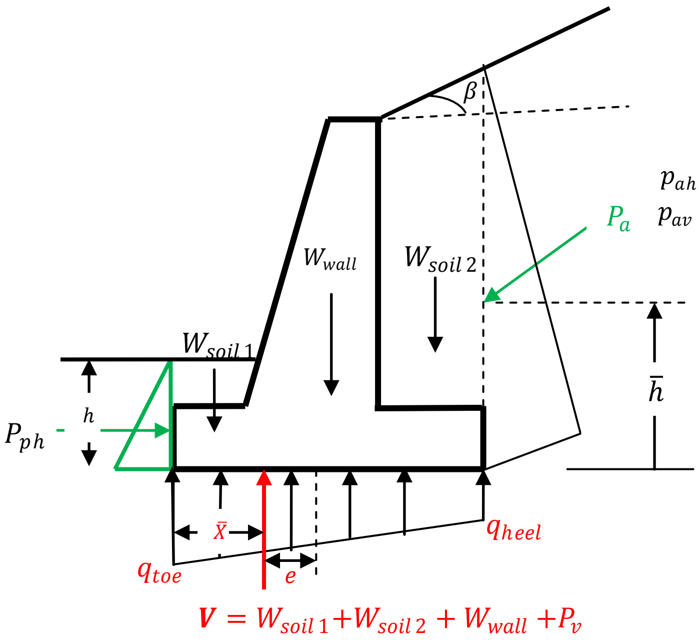


Figure 3.3: Forces on retaining wall

a) Overturning

Resisting Moment = Moment of W_{wall} , W_{soil} *and* P_{av} about the toe

Overturning Moment = Moment of P_{ah} about the toe

FS_{ot} = Factor of safety against overturning

For stability against overturning; $\frac{\text{Resisting Moment}}{\text{Overturning Moment}} \geq FS_{ot}$ (3.1)

b) Sliding

Base soil parameters: c_w, δ

$$\boldsymbol{F_r} = V \tan\delta + c_w B + P_{ph} \qquad (3.2)$$

FS_{sl} = Factor of safety against sliding

For stability against sliding; $\frac{\boldsymbol{F_r}}{\boldsymbol{P_{ah}}} \geq \boldsymbol{FS_{sl}}$ (3.3)

c) Bearing Capacity

Measured from the toe the resultant passes at $\bar{X}$ distance;

$$\bar{X} = \frac{Resisting\ Moment - Overturning\ Moment}{v} \quad (3.4)$$

Measured from the geometric center of the base, the resultant passes at e distance;

$$e = \left|\frac{B}{2} - \bar{X}\right| \quad (3.5)$$

To avoid over stressing at the toe or heel, e should be in the middle third, i.e, $e \leq \frac{B}{6}$

The pressure intensity under the soil is calculated as:

$$q = \frac{V}{B}\left[1 \pm \frac{6e}{B}\right] \quad 3.6a$$

$$q_{max} = \frac{V}{B}\left[1 + \frac{6e}{B}\right] \quad 3.6b$$

$$\boldsymbol{q_{min}} = \frac{\boldsymbol{V}}{\boldsymbol{B}}\left[\mathbf{1} - \frac{\mathbf{6}\boldsymbol{e}}{\boldsymbol{B}}\right] \quad 3.6c$$

let $q_{all} = allowable\ bearing\ capacity$

For stability against bearing capacity failure; $\boldsymbol{q_{all}} \geq \boldsymbol{q_{max}}$

4. APPLICATION OF THE SOFTWARE

In this chapter, the contents of the developed software and the procedures that will be followed to use it are presented.

4.1. General

When the program is launched, the window in Figure 4.1 will appear. The window contains all the controlling tools typical for the Windows environment (minimizing, maximizing and closing the application window).

Moreover, the window contains menus at its upper left corner and tabs at its bottom left side. The menus are File, Define, Analysis, Retaining wall, Output and Help and the tabs are Theory, Ground, Geometry, Profile, Soils, Assign, Ground water, Surcharge, Analysis and Retaining wall). The contents of these menus and tabs are given in section 4.2 below.

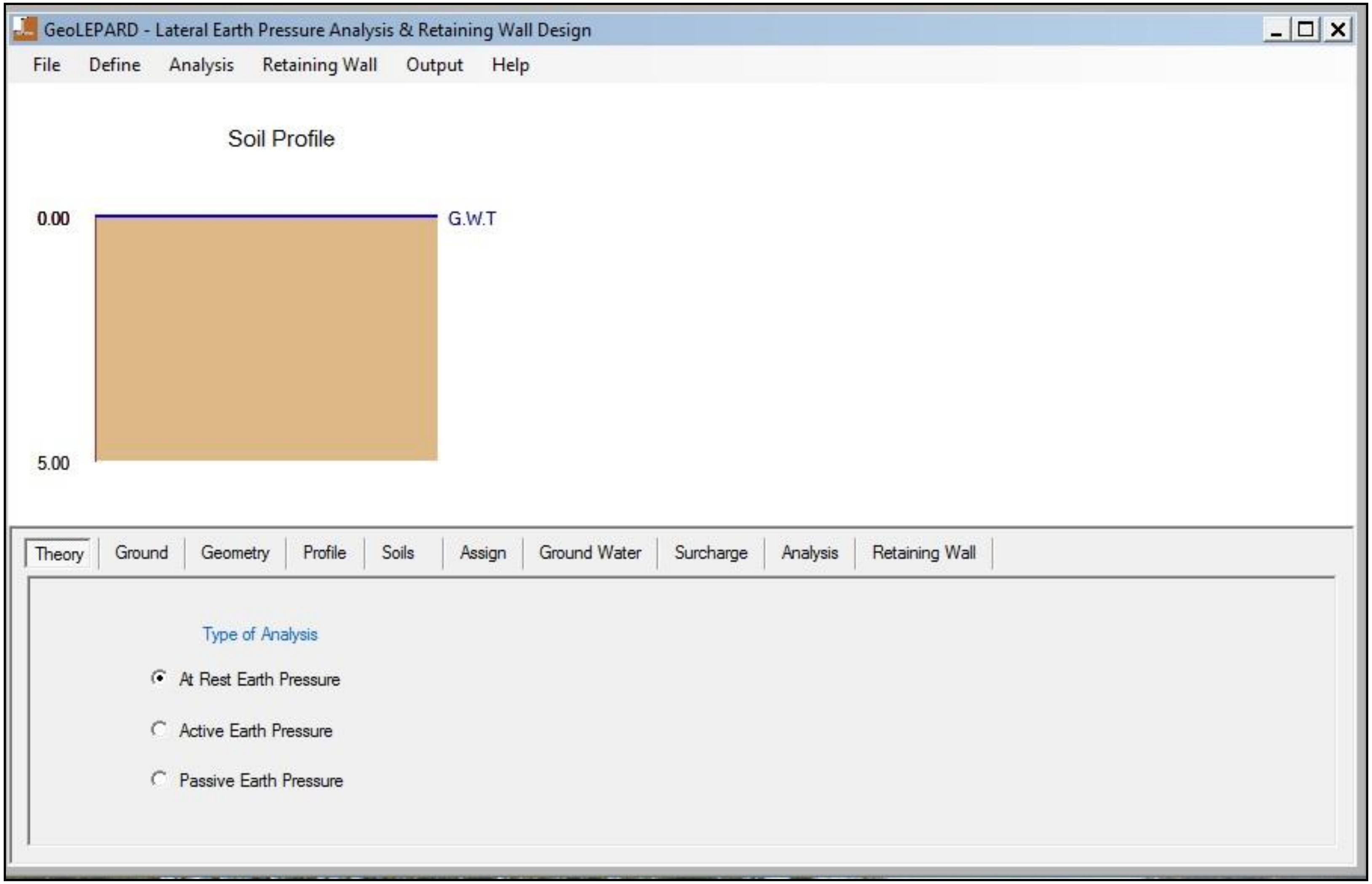


Figure 4.1: Main Window

4.2. Menus and Tabs

4.2.1. Menus

- **File**: this menu contains New, Open, Save, Save As and Exit commands. The New command lets the user to launch the software afresh. The Open commend is used to open previously saved projects. The Save commend saves an update to an existing project. The Save As command saves a project in XML file format. The Exit command terminates the software.

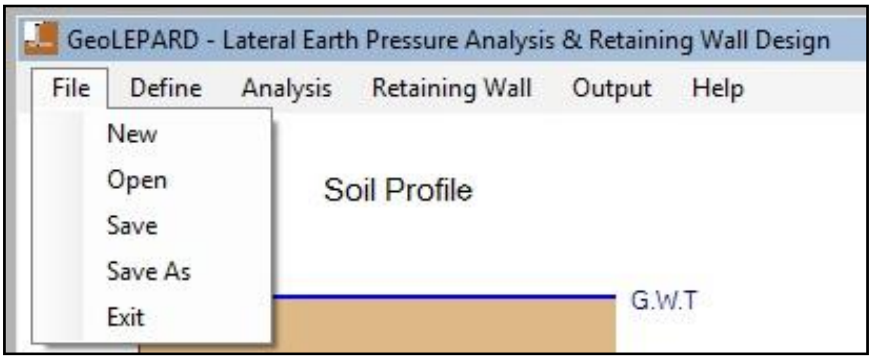


Figure 4.2: File menu

- **Define**: this menu contains the commands to let the user select a tab at the bottom of the window.

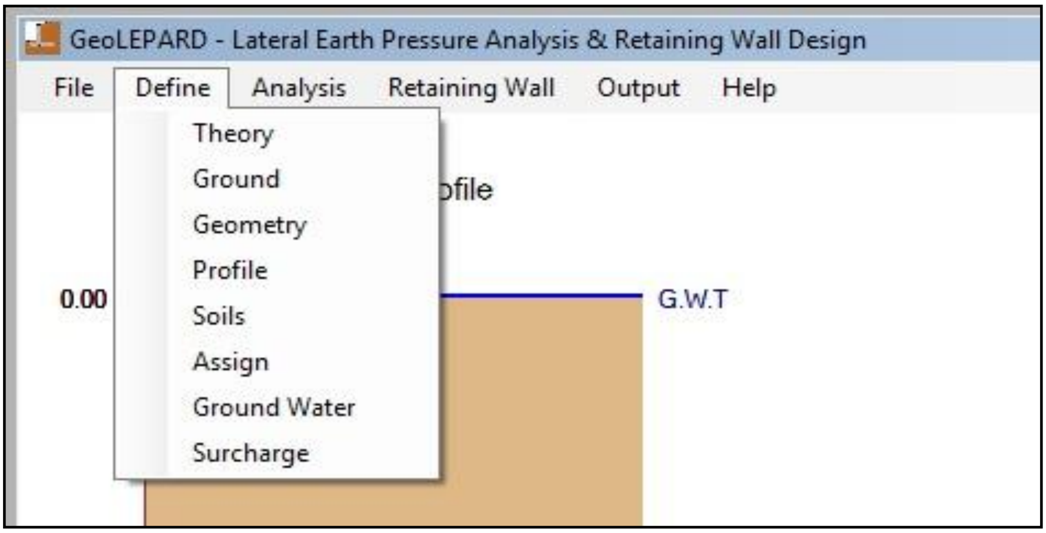


Figure 4.3: Define menu

- **Analysis:** this menu contains the commands to let the user select the analysis tab at the bottom of the window and run the analysis.

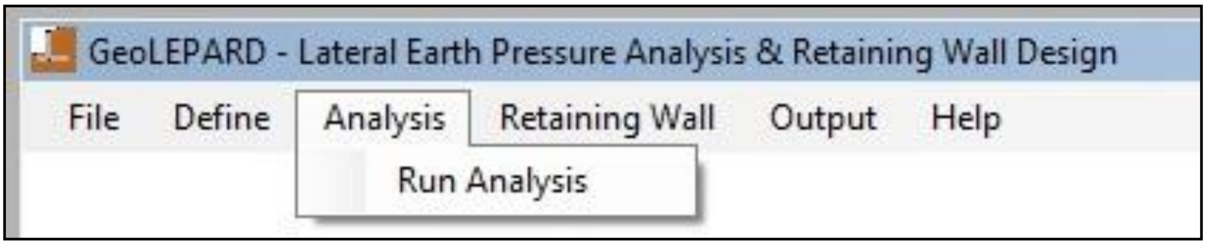


Figure 4.4: Analysis Menu

- **Retaining wall:** this menu contains the command, "View/Edit Parameters", to let the user select the retaining wall tab and view the retaining wall window.

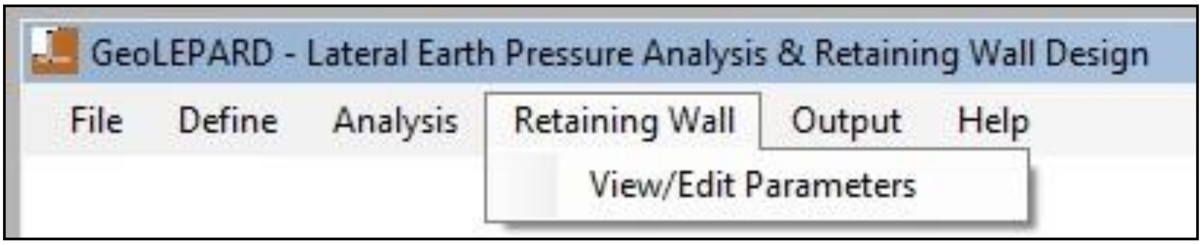


Figure 4.5: Retaining Wall Menu

- **Output:** this menu contains the "Show Calculation Table" command. The command prompts the program to show calculation table for earth pressure coefficients and analysis of earth pressure and the input values.

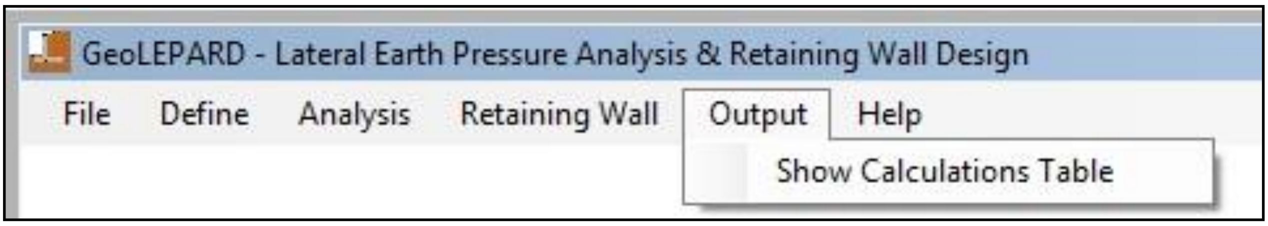


Figure 4.6: Output menu

- **Help:** this menu displays information about the software and provides access to the help files of the software.

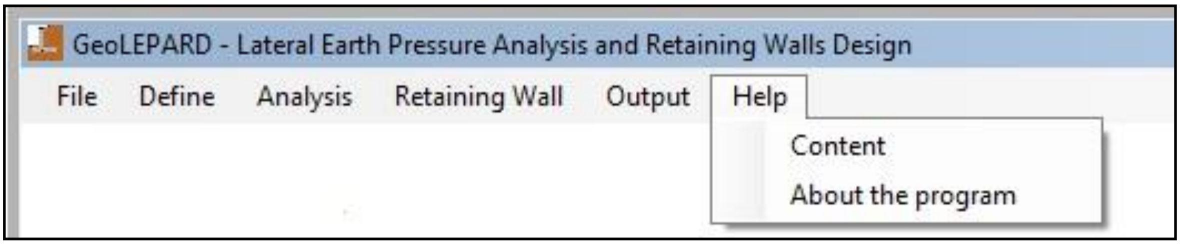


Figure 4.7: Help Menu

4.2.2. Tabs

1. **Theory:** this tab lets the user select the type of analysis and the earth pressure theory for the computation of lateral earth pressure.(See Fig. 4.8 below)

2. **Ground**: this allows the user to specify the ground (backfill) inclination angle. (See Fig. 4.9 below)

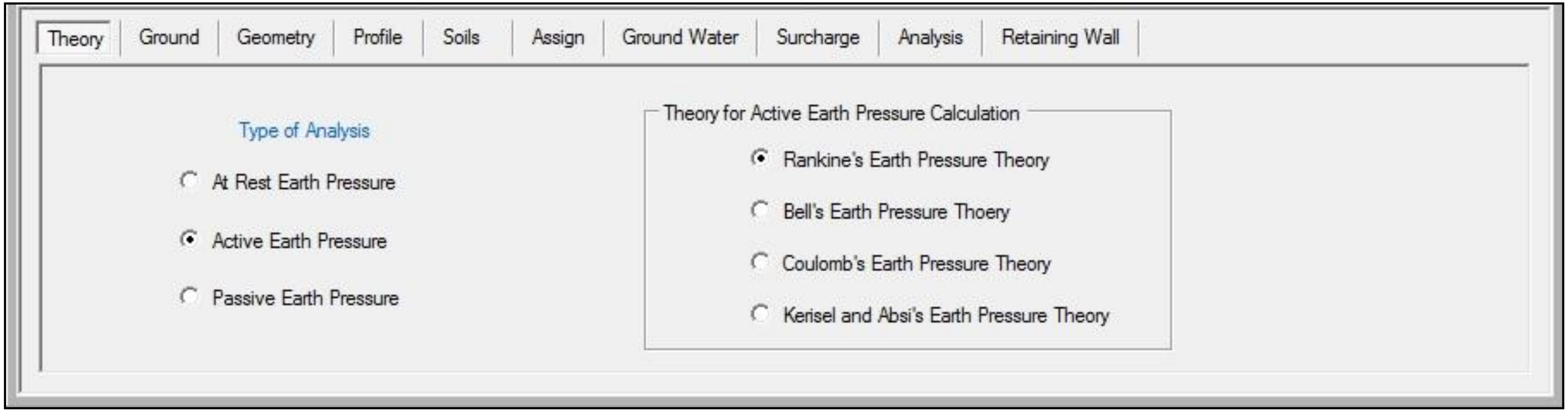


Figure 4.8: Theory tab

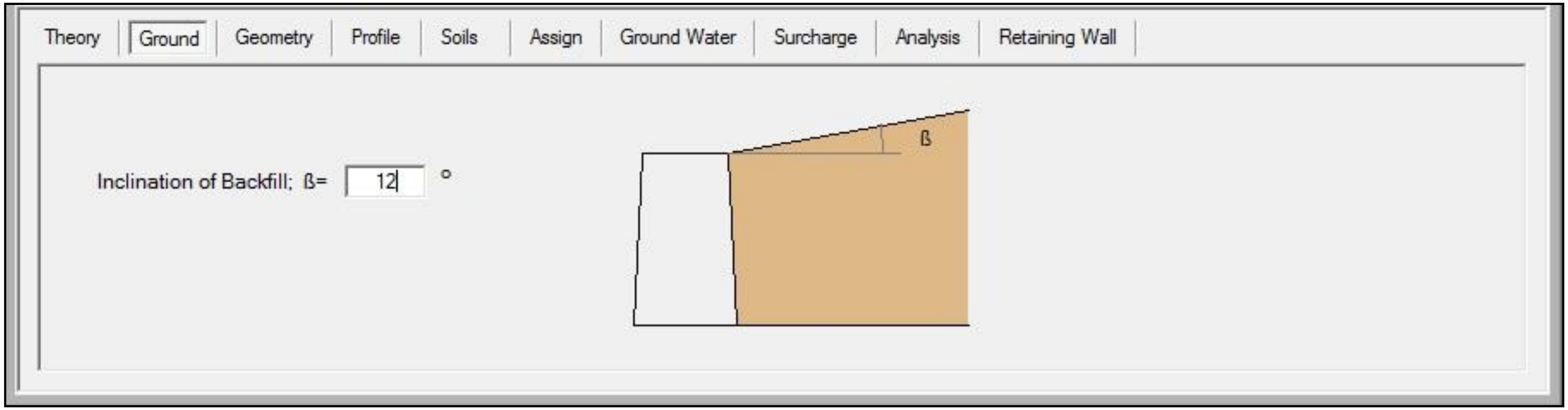


Figure 4.9: Ground tab

3. **Geometry**: this allows the user to specify the inclination angle of the back face of the retaining wall.(See Fig. 4.10 below)

4. **Profile:** lets the user to input the height of the retaining wall and interfaces. After specifying the height and interfaces it is possible to edit/remove individual layers with the help of the edit/remove buttons. (See Fig. 4.11 below)

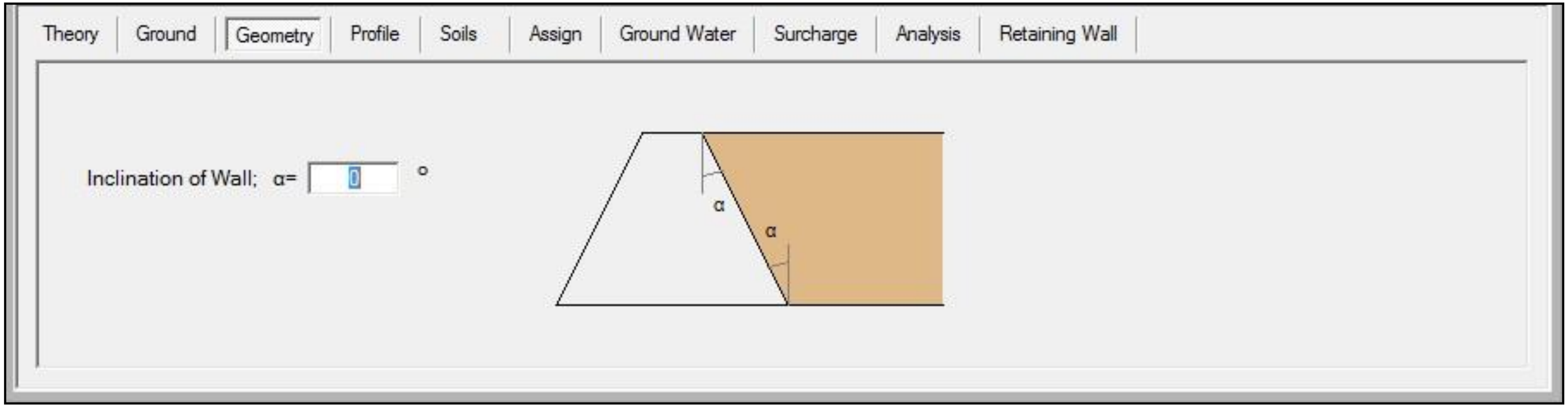
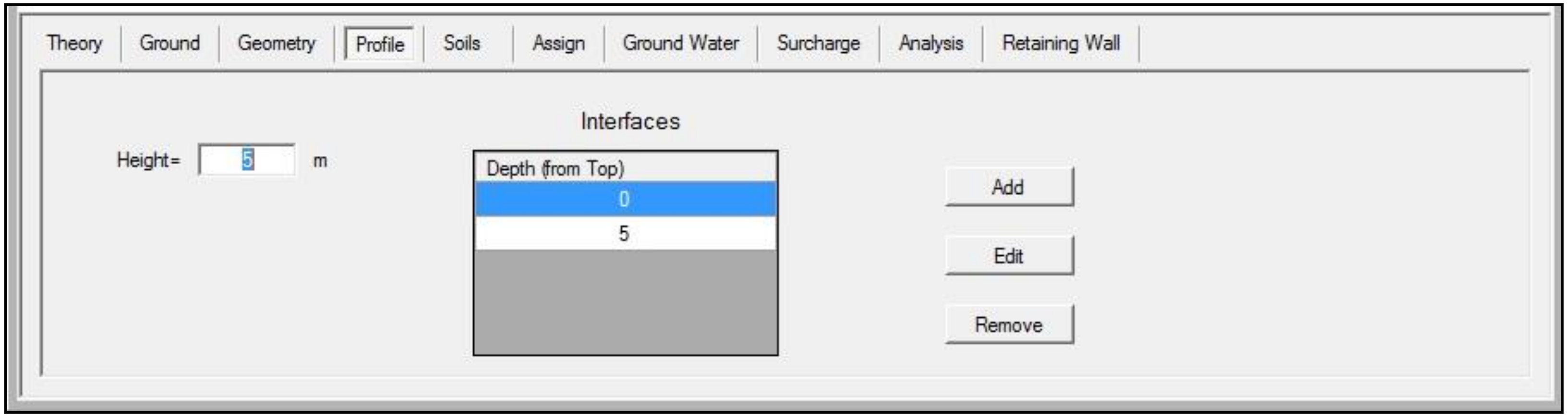
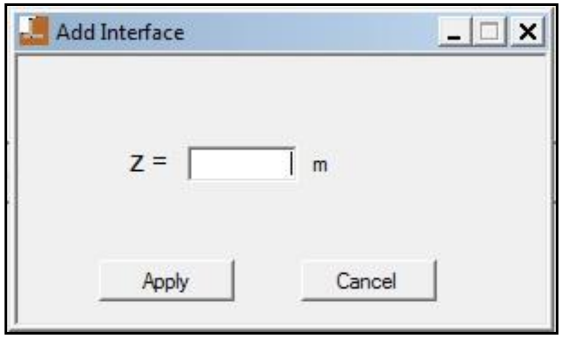


Figure 4.10: Geometry tab



(a)

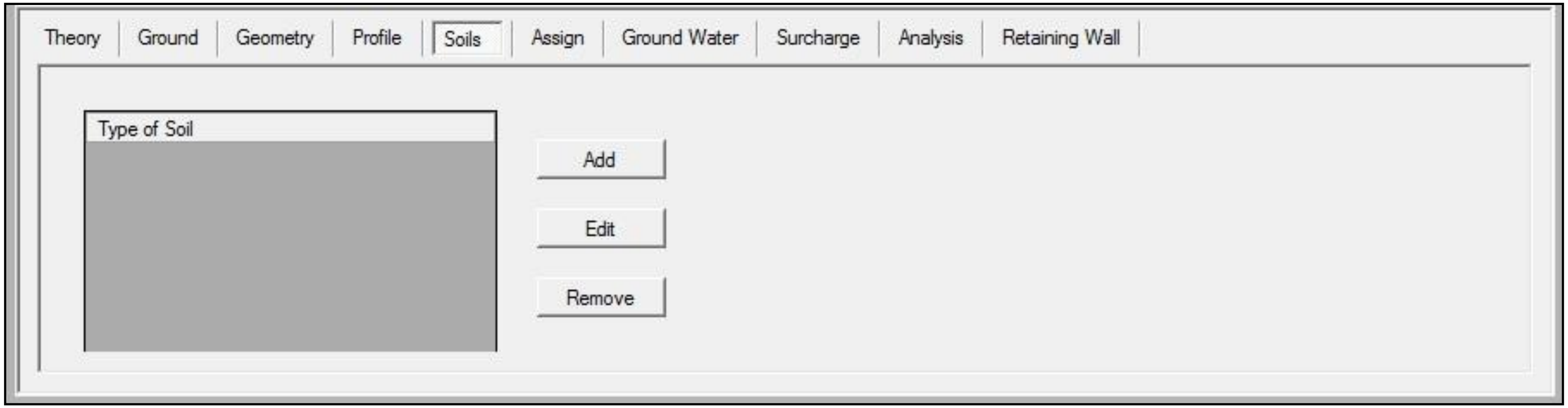


(b)

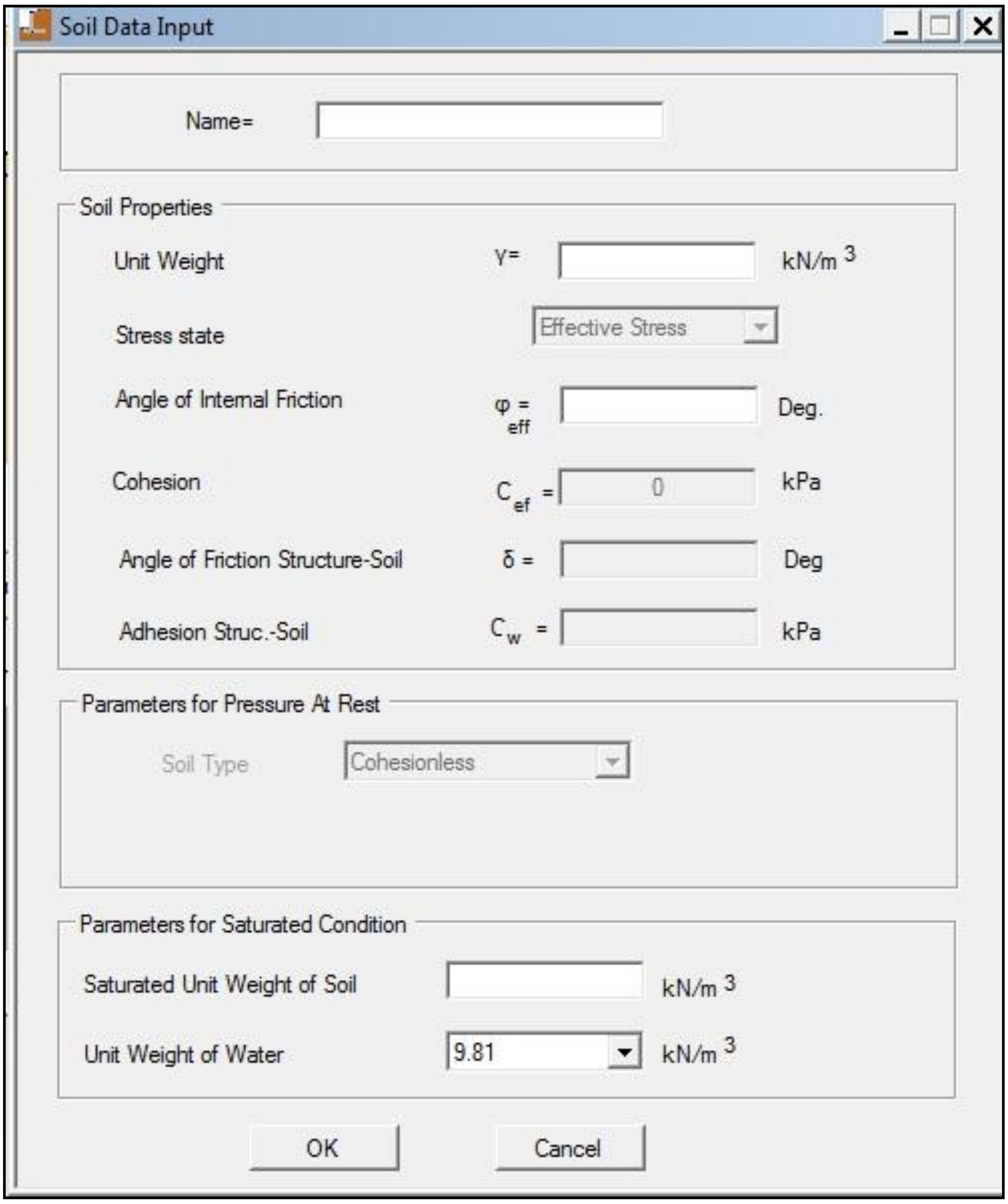
Figure 4.11: (a) Profile tab (b) “Add interface” dialogue box

5. **Soils:** lets the user input, edit and remove soil characteristics of the different layers. Adding (editing) a soil is performed in the “Soil Data Input” dialogue box. This part of the window serves to introduce basic parameters of soils – unit weight, angle of internal friction and cohesion.

   Either *effective or total* stress state parameters of the angle of internal friction and cohesion are inserted depending on the setting in the "**Stress state**" combo list. One further needs to specify the angle of internal friction between the soil and structure, which depends on the structure material and the type of soil.


Theory
Ground
Geometry
Profile
Soils
Assign
Ground Water
Surcharge
Analysis
Retaining Wall
Type of Soil
Add
Edit
Remove

(a)



(b)

Figure 4.12: (a) Soils tab (b) Soil Data Input dialogue box

6. **Assign**: contains a list of layers of profile and their respective thickness. Here each layer is associated with list of predefined soils. The list of soils is accessible from a combo list for each layer of the profile. In order to assign a soil, it requires one to select soil type from the combo list.

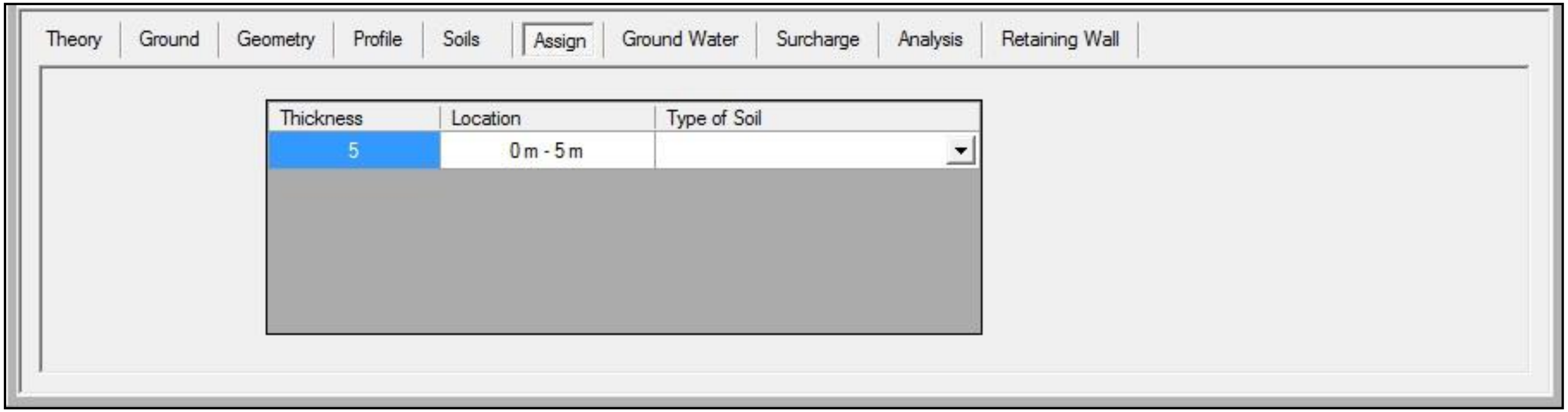


Figure 4.13: Assign tab

7. **Ground Water:** allows the user to define the depth of water level. As the user inputs a non-zero value, the water level is displayed graphically as shown in figure 4.14.

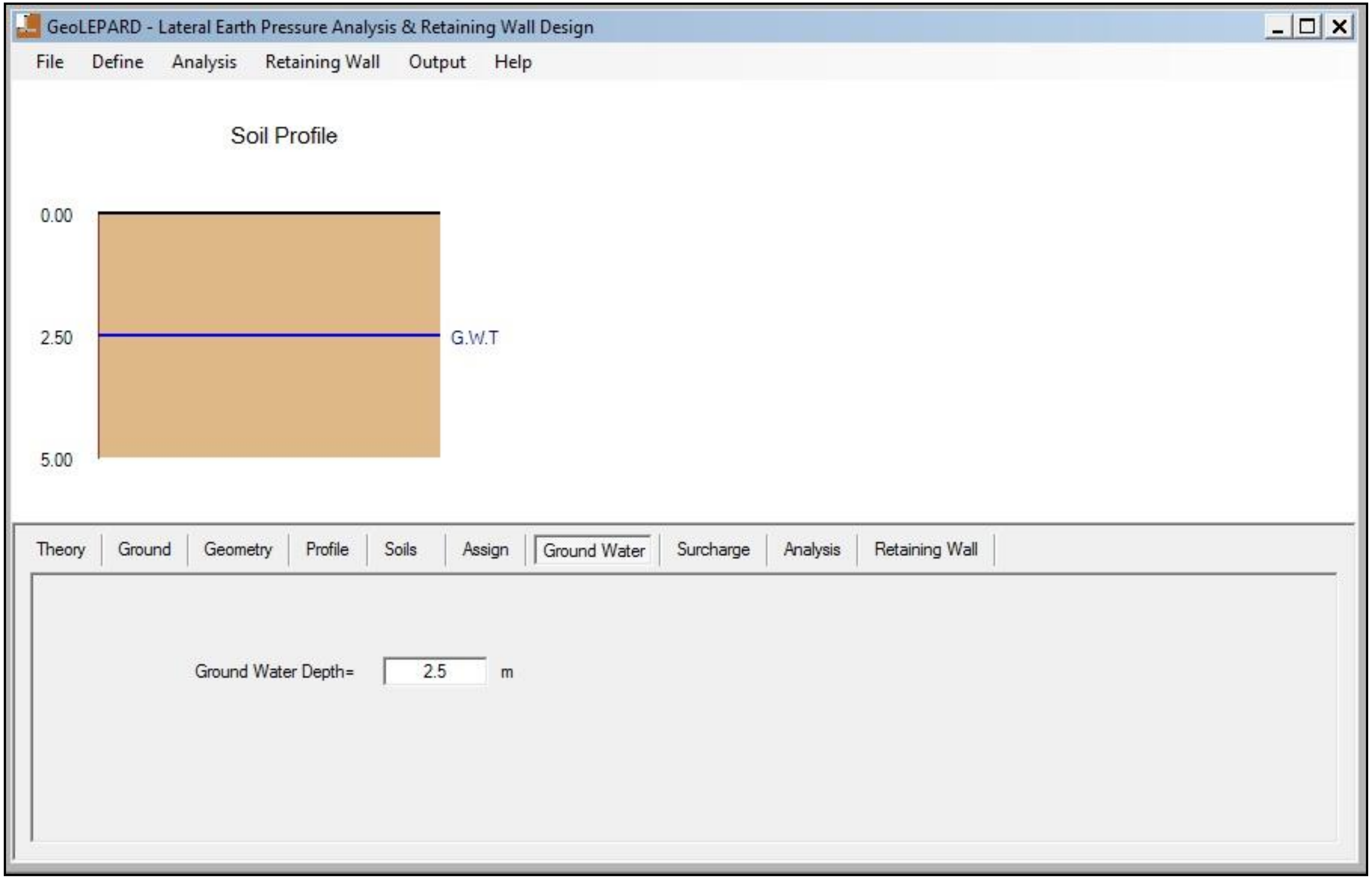


Figure 4.14: Water tab and graphical display

8. **Surcharge:** allows the user to input/edit the magnitude of uniform surcharge load. As the user enters a non-zero value, the surcharge is displayed graphically as shown in figure 4.15.

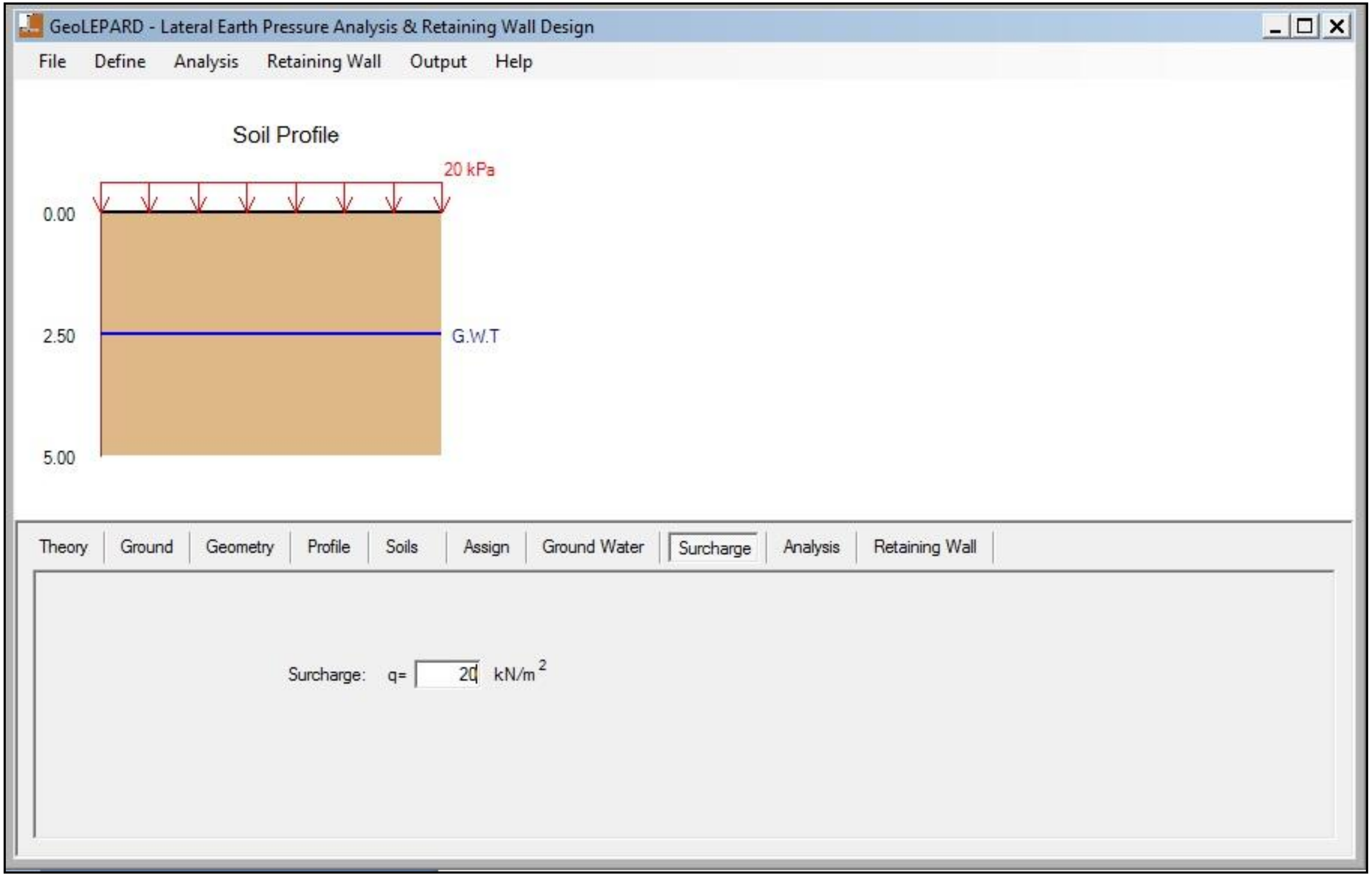


Figure 4.15: Surcharge tab and graphical display

9. **Analysis:** This serves to run the analysis for each type of earth pressure (active pressure, pressure at rest, passive pressure) and earth pressure theory (Rankine(Bell), Coulomb and Kerisel & Absi).

   The analysis results are displayed on the desktop and are updated for an arbitrary change in input data.

   Moreover, it gives the user the option of computing hydrostatic pressure due to tension cracks filled with water.

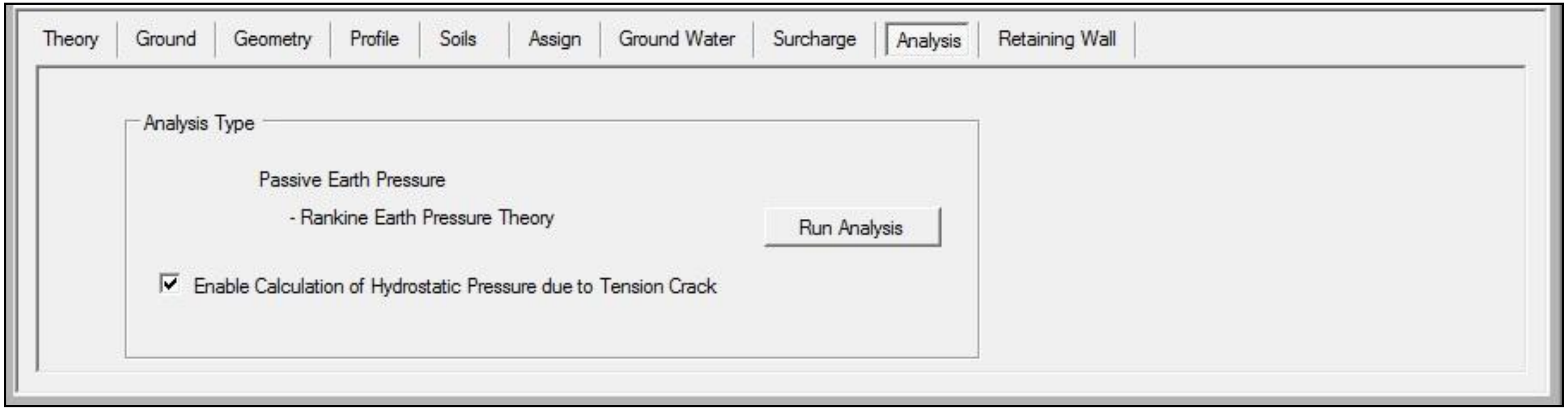


Figure 4.16: Analysis tab

10. **Retaining Wall:** allows the user to select the type of retaining wall to be designed (Gravity/Cantilever) and edit the default dimensions given in the text boxes.

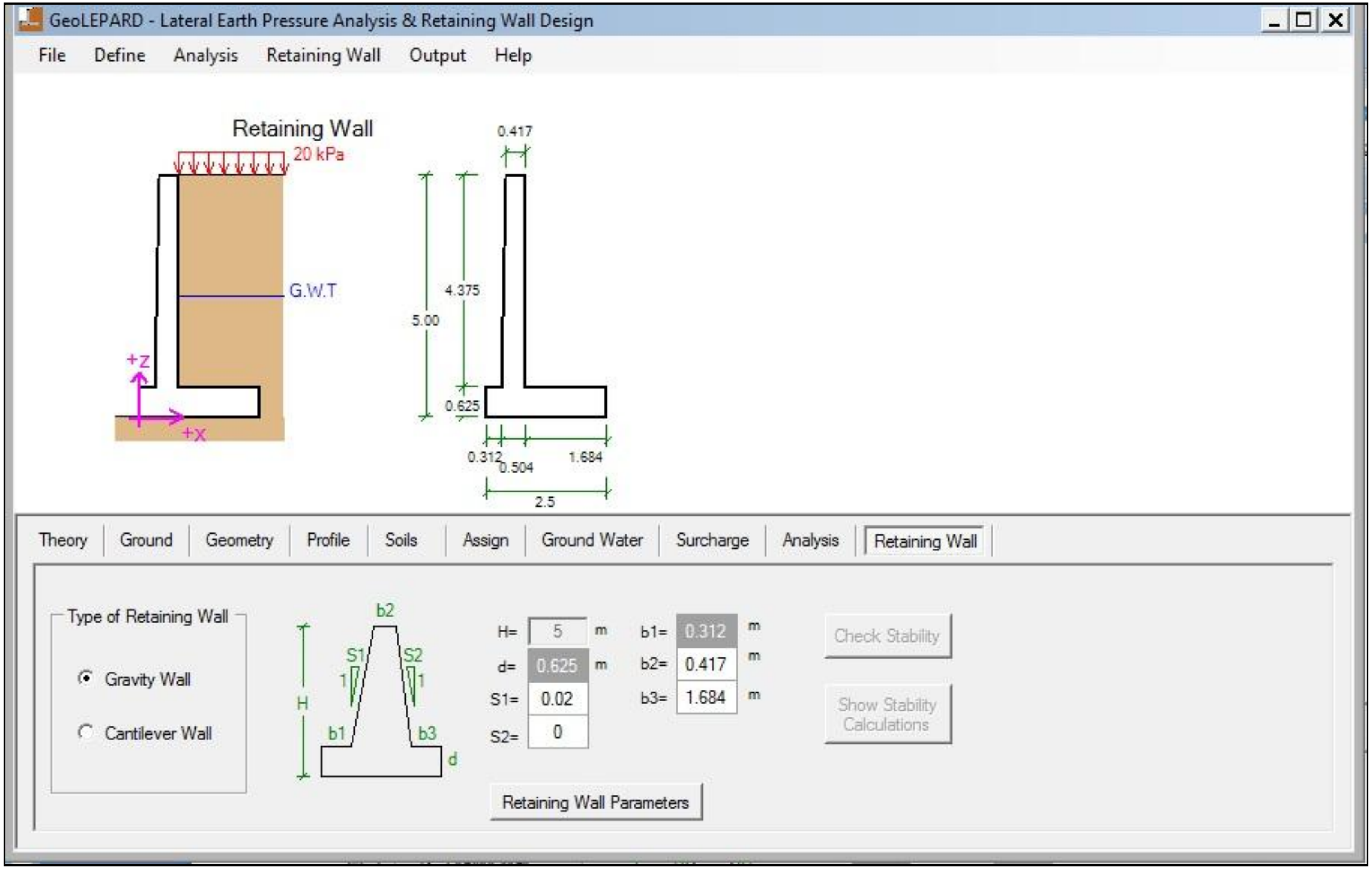


Figure 4.17: Retaining Wall tab

The necessary data used for the design/analysis of the retaining walls are entered using the "Retaining wall Parameters" dialogue box. Moreover, the analysis is run by clicking the "Check Stability" button and calculations can be shown clicking the "Show Stability Calculations" button. The analysis results are displayed graphically on the desktop and are updated for an arbitrary change in input data.

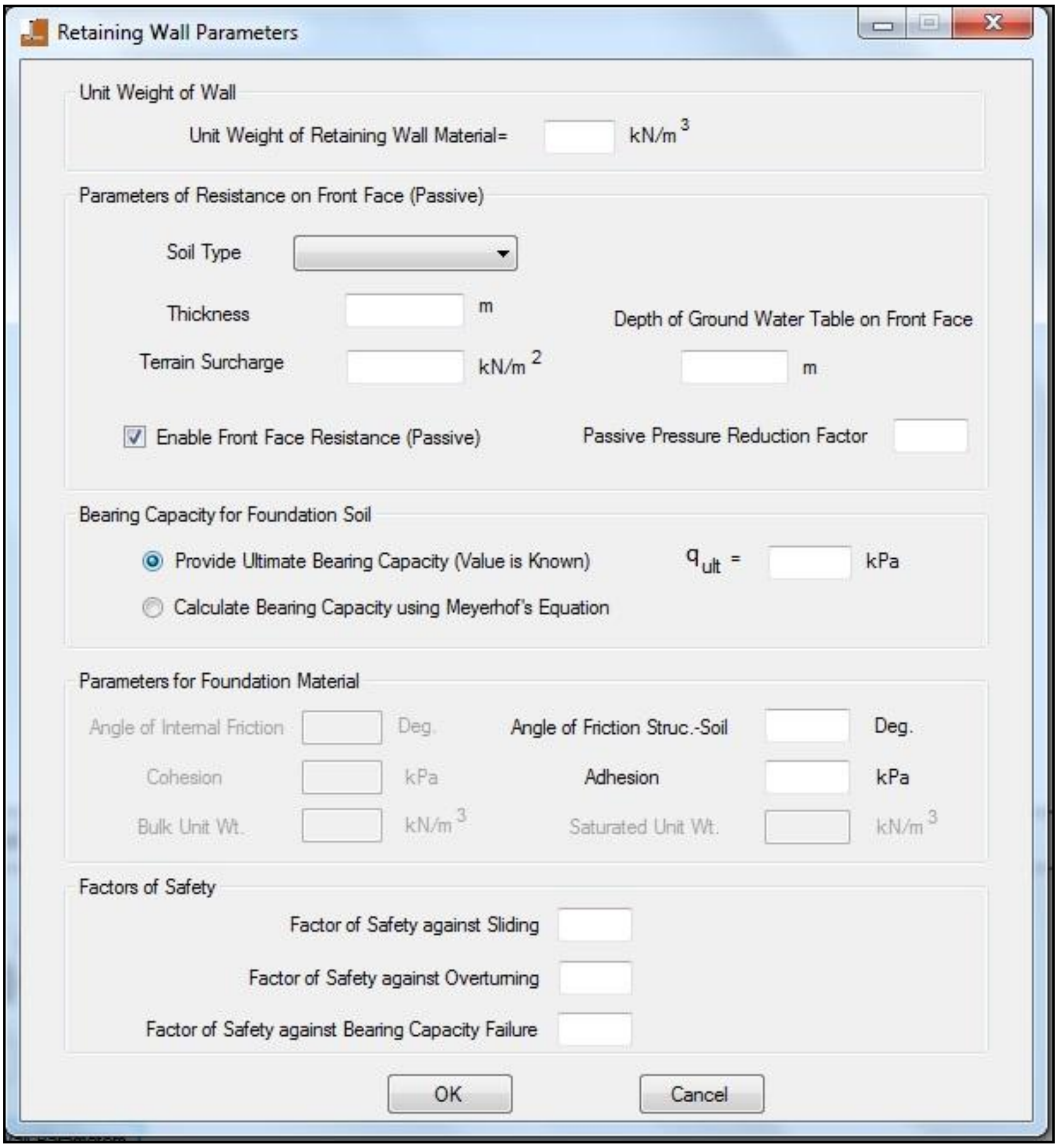


Figure 4.18: Retaining Wall Parameters dialogue box

4.3. Analysis of At-rest Earth Pressure

To execute this analysis one needs to go through the following steps;

1. Run the software and click the "Theory" tab and select the "At Rest Earth Pressure" radio button (Figure 4.8).

2. Click the "Ground" tab and enter the angle of inclination of the back fill (ground) to be retained in degrees (Figure 4.9).

3. Click the "Geometry" tab and enter the angle the wall makes with the vertical in degrees (Figure 4.10).
4. Click the "profile" tab and enter the height of the retaining wall then click the "Add" button and input the interfaces in the "Add interface" dialogue box and click "Apply" (Figure 4.11).

5. Click the "Soils" tab and then the "Add" button and enter the necessary soil data in the "Soil Data Input" dialogue box and click "OK" (Figure 4.12).

6. Click the "Assign" tab and then open the "Type of Soil" combo list for each layer of the profile in order to assign soil (Figure 4.13).

7. Click the "Ground Water" tab and enter the depth of water level measured from the top (Figure 4.14).

8. Click the "Surcharge" tab and enter the magnitude of the uniform surcharge load (Figure 4.15).

9. Click the "Analysis" tab and check/uncheck the soil wedge computation check box depending on your interest then click the "Run Analysis" button. A graphical and tabular output like the one shown in figure 4.19 appears at the desktop.

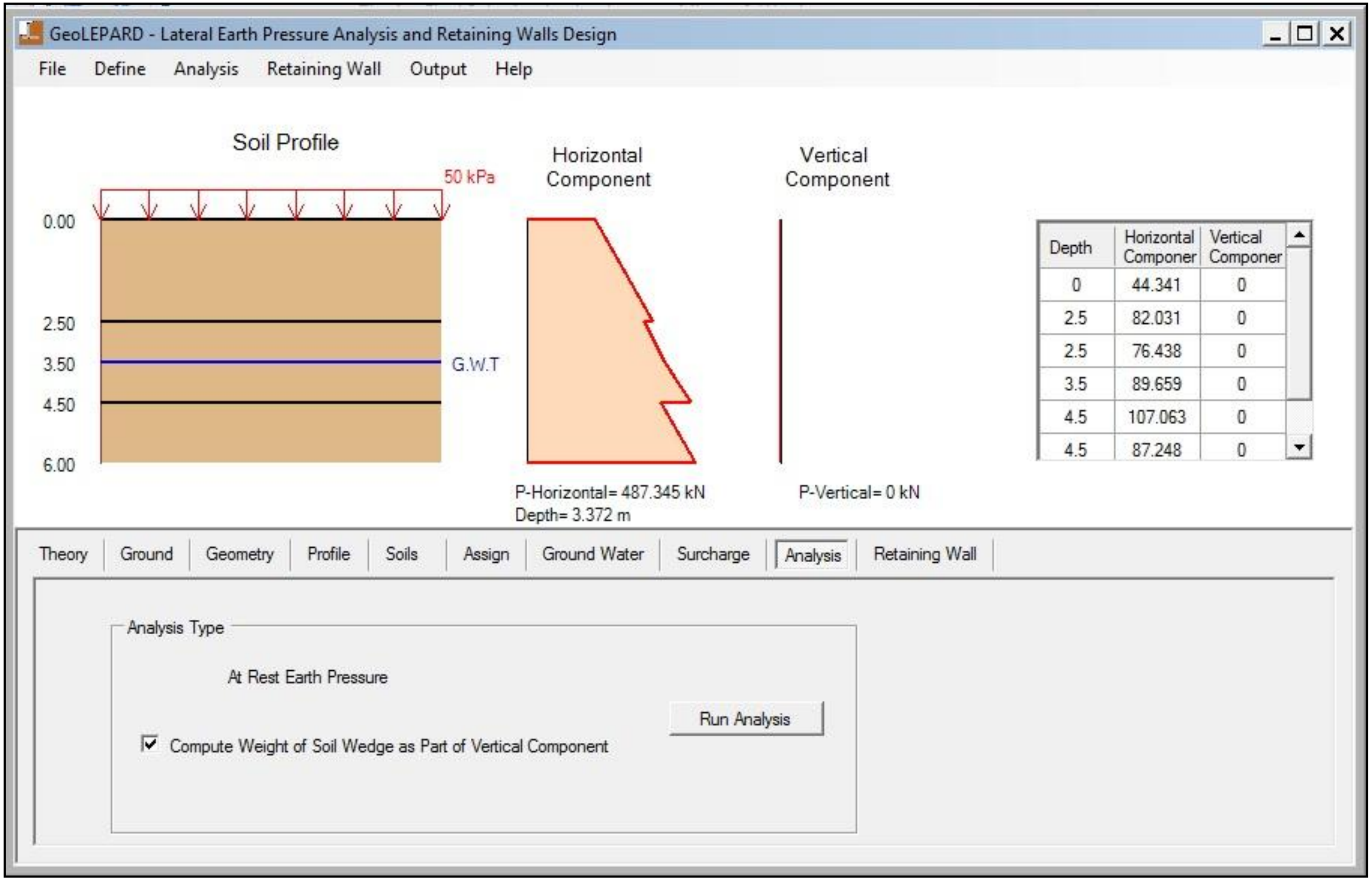


Figure 4.19: Graphical and tabular output of at-rest earth pressure analysis

4.4. Analysis of Active Earth Pressure

4.4.1. Rankine Earth Pressure Theory

To execute this analysis the following series of steps shall be followed;

1. Click the “Theory” tab and then select the “Active Earth Pressure” radio button. Then from the “Theory for Active Earth Pressure Calculation” group box select the “Rankine Earth Pressure Theory” radio button (Figure 4.20).

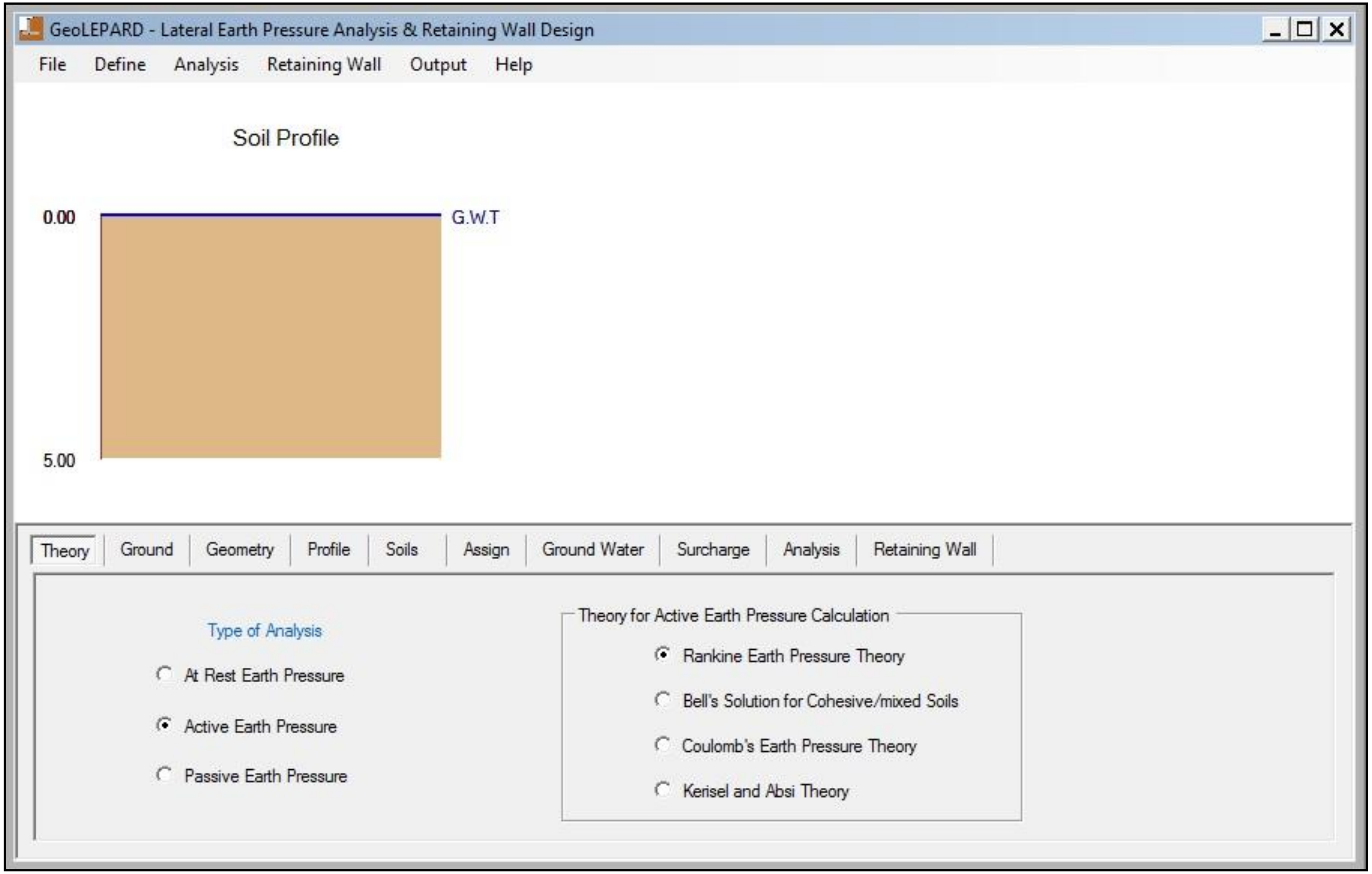


Figure 4.20: Rankine Earth Pressure Theory selection

2. Repeat the steps (step 2- step 8) in section 4.3.

3. Click the “Analysis” tab, check/uncheck to incorporate/omit computation of hydrostatic pressure due to tension crack filled with water. And then click the “Run Analysis” button. A graphical and tabular output like the one shown in figure 4.21 appears at the desktop.

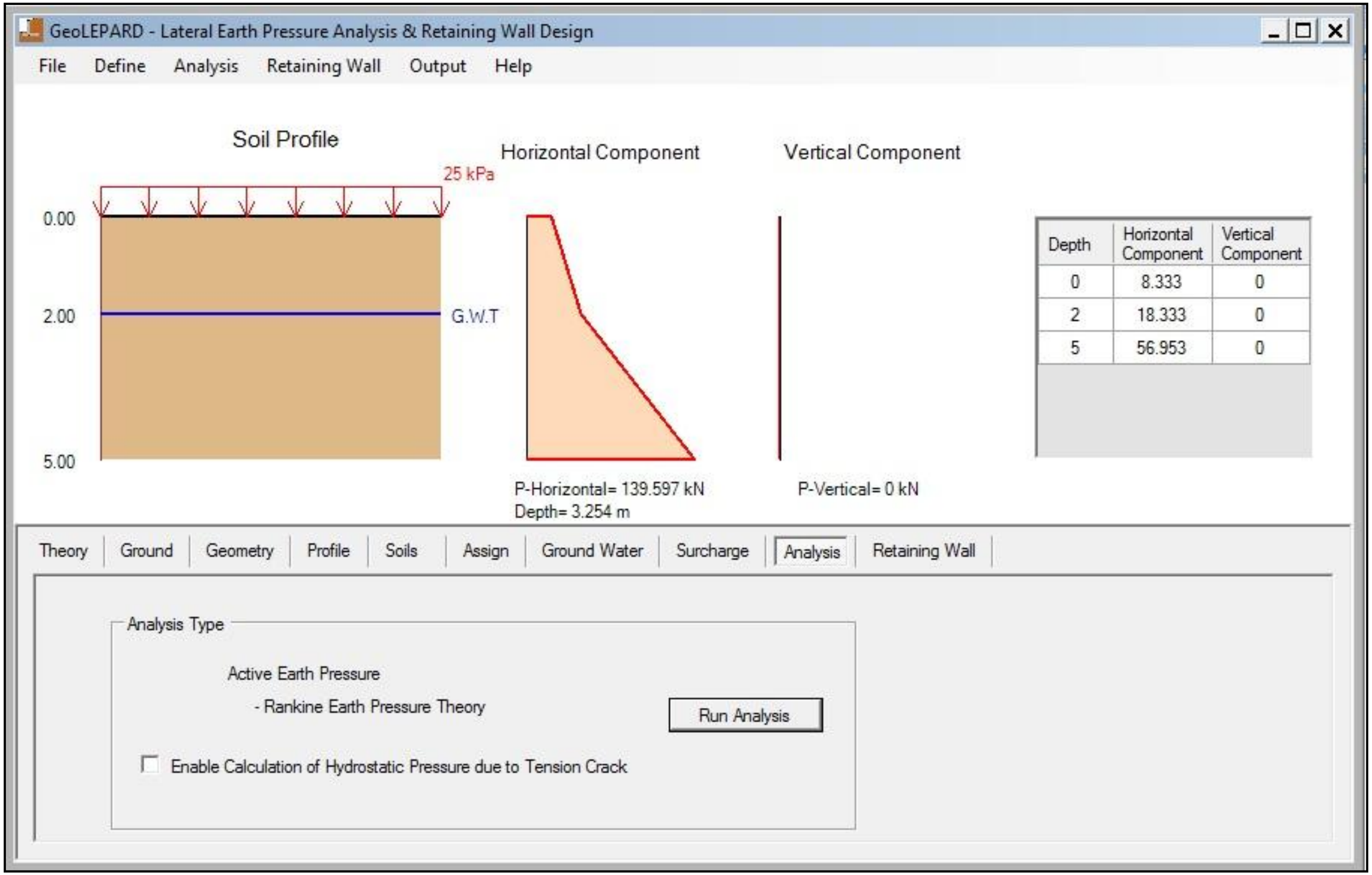


Figure 4.21: Graphical and tabular output of Rankine earth pressure analysis

4.4.1.1. Bell's Solution for Cohesive/Mixed Soils

1. Click the "Theory" tab and then select the "Active Earth Pressure" radio button. Then from the "Theory for Active Earth Pressure Calculation" group box select the "Bell's Solution for Cohesive/Mixed Soils" radio button.

2. Repeat the steps 2 and 3 in section 4.4.1.

4.4.2. Coulomb's Earth Pressure Theory

1. Click the "Theory" tab and then select the "Active Earth Pressure" radio button. Then from the "Theory for Active Earth Pressure Calculation" group box select the "Coulomb's Earth Pressure Theory" radio button.

2. Repeat steps 2 and 3 in section 4.4.1. Then a graphical and tabular output like the one shown in figure 4.22 shall be displayed.

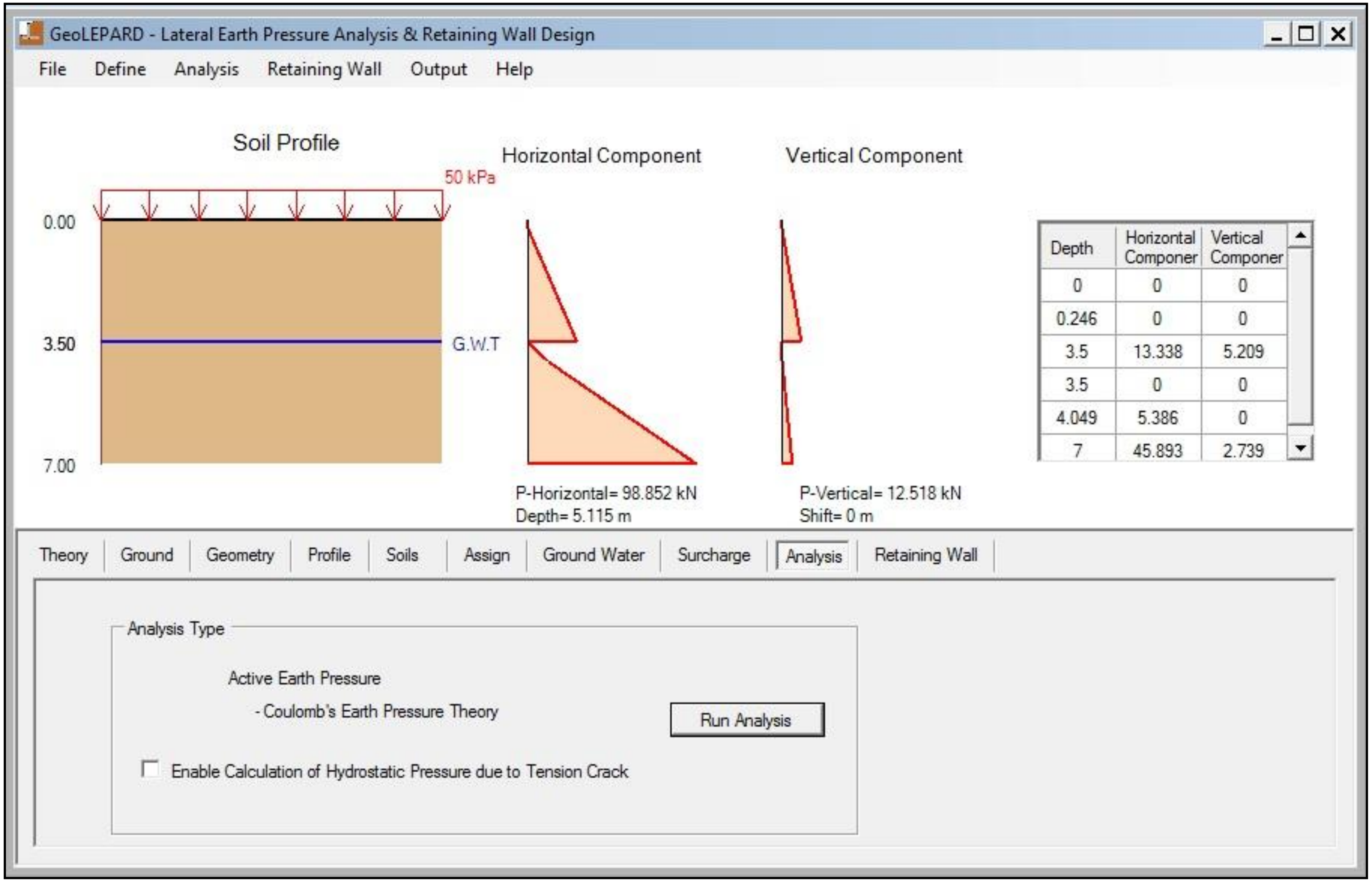


Figure 4.22: Graphical and tabular output of Coulomb's earth pressure analysis

4.4.3. Kerisel and Absi Earth Pressure Theory

To execute this analysis one needs to go through the following steps;

1. Click the “Theory” tab and then select the “Active Earth Pressure” radio button. Then from the “Theory for Active Earth Pressure Calculation” group box select the “Kerisel and Absi Earth Pressure Theory” radio button.

2. Repeat steps 2 through 4 in section 4.3.

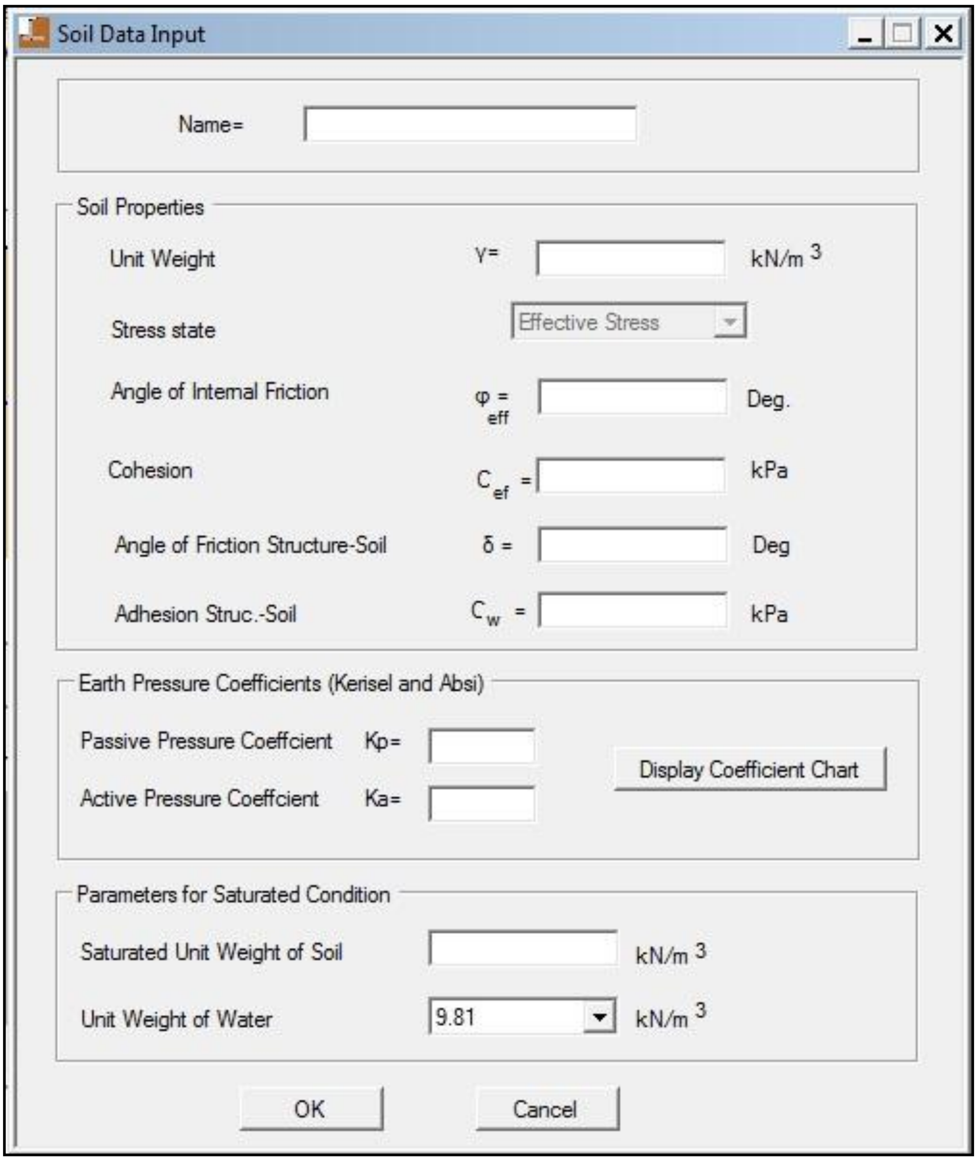


Figure 4.23: Soil Data Input dialogue box for Kerisel and Absi Theory

3. Click the “Soils” tab and then the “Add” button and enter the necessary soil data in the “Soil Data Input” dialogue box. To input the “Earth Pressure Coefficient” click the “Display Coefficient Chart” button and use the Kerisel & Absi chart (Figure 2.23). When done with entering the data click “OK” (Figure 4.23).

4. Repeat steps 6 through 8 in section 4.3.

5. Click the “Analysis” tab and then the “Run Analysis” button. A graphical and tabular output like the one shown in figure 4.24 appears at the desktop.

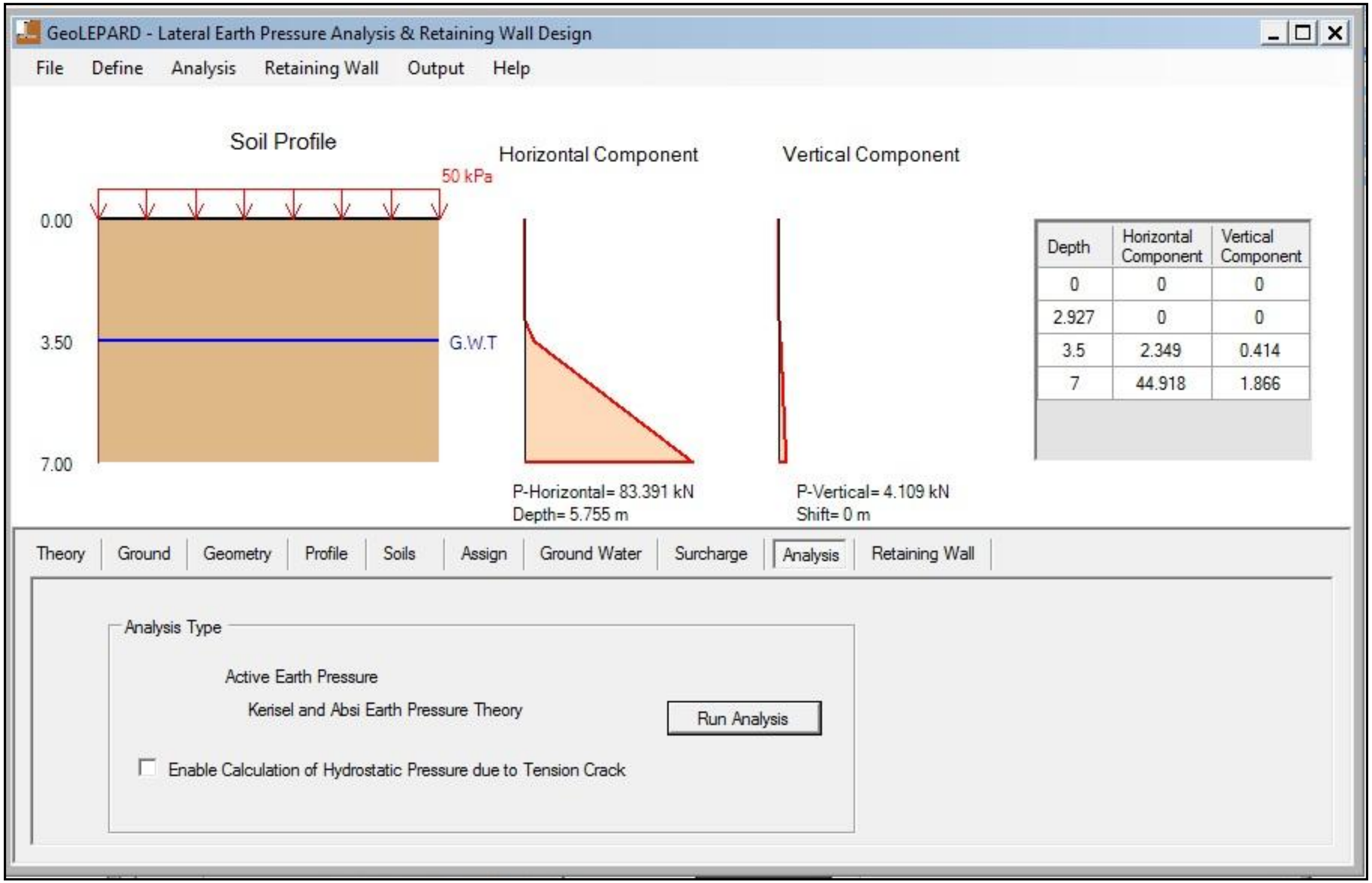


Figure 4.24: Graphical and tabular output of Coulomb's earth pressure analysis

4.5. Analysis of Passive Earth Pressure

To come up with the result of the analysis for Passive Earth Pressure the same steps, except selection of "Type of Analysis", as that of section 4.4 shall be followed.

4.6. Retaining Walls

The software designs/analyzes Gratvity and Cantelever retaining walls for active earth pressure case at the back face of the wall and for passive earth pressure at the front face. Having gone through all the necessary steps described in section 4.4, one can design/analyze retaining walls at the "Retaining Wall" tab.

4.6.1. Design/Analysis of Retaining Walls

For the design/analysis of retaining walls the following series of steps shall be followed.

1. Click the "Retaining Wall" tab.

2. From the "Type of Retaining Wall" group box, select one of the two radio buttons to choose between Gravity or Cantilever wall. For each selection, the dimension text boxes display default dimensions, i.e., the minimum of the tentative dimension range given in section 2.7. The user can edit these dimensions in to values of his interest (Figure 4.17).

3. Click the "Retaining Wall Parameters" button and input the parameters necessary for the design/analysis of the retaining wall in the "Retaining Wall Parameters" dialogue box (Figure 4.18). When done with filling out the box click "OK" and the "Check Stability" button will be active (Figure 4.16).

4. To check the stability of the wall at hand, click the "Check Stability" button. Now graphical and tabular form of results are displayed at the desktop (Figure 4.25).Hereafter the "Show Stability Calculations" button will be active.

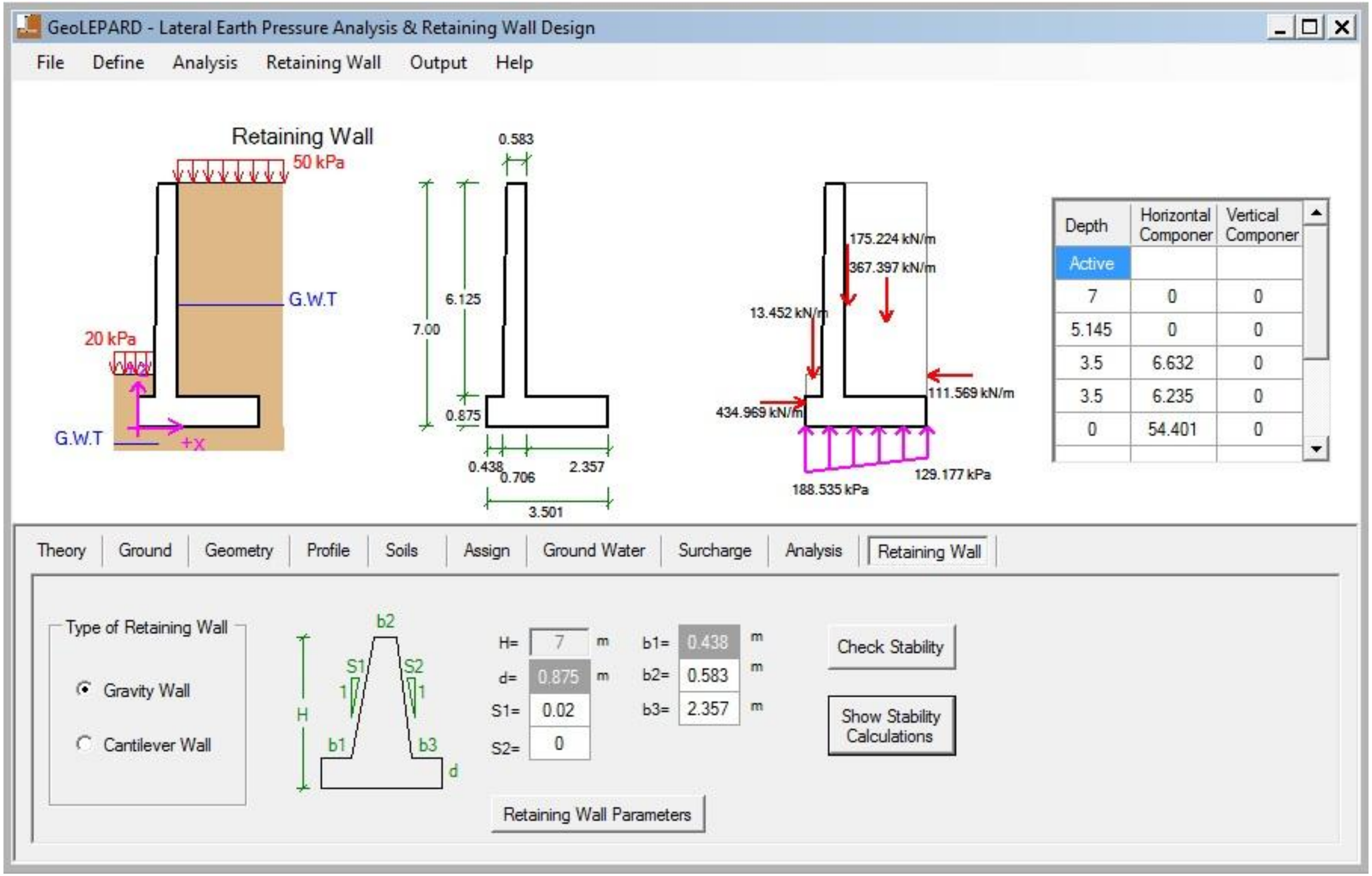
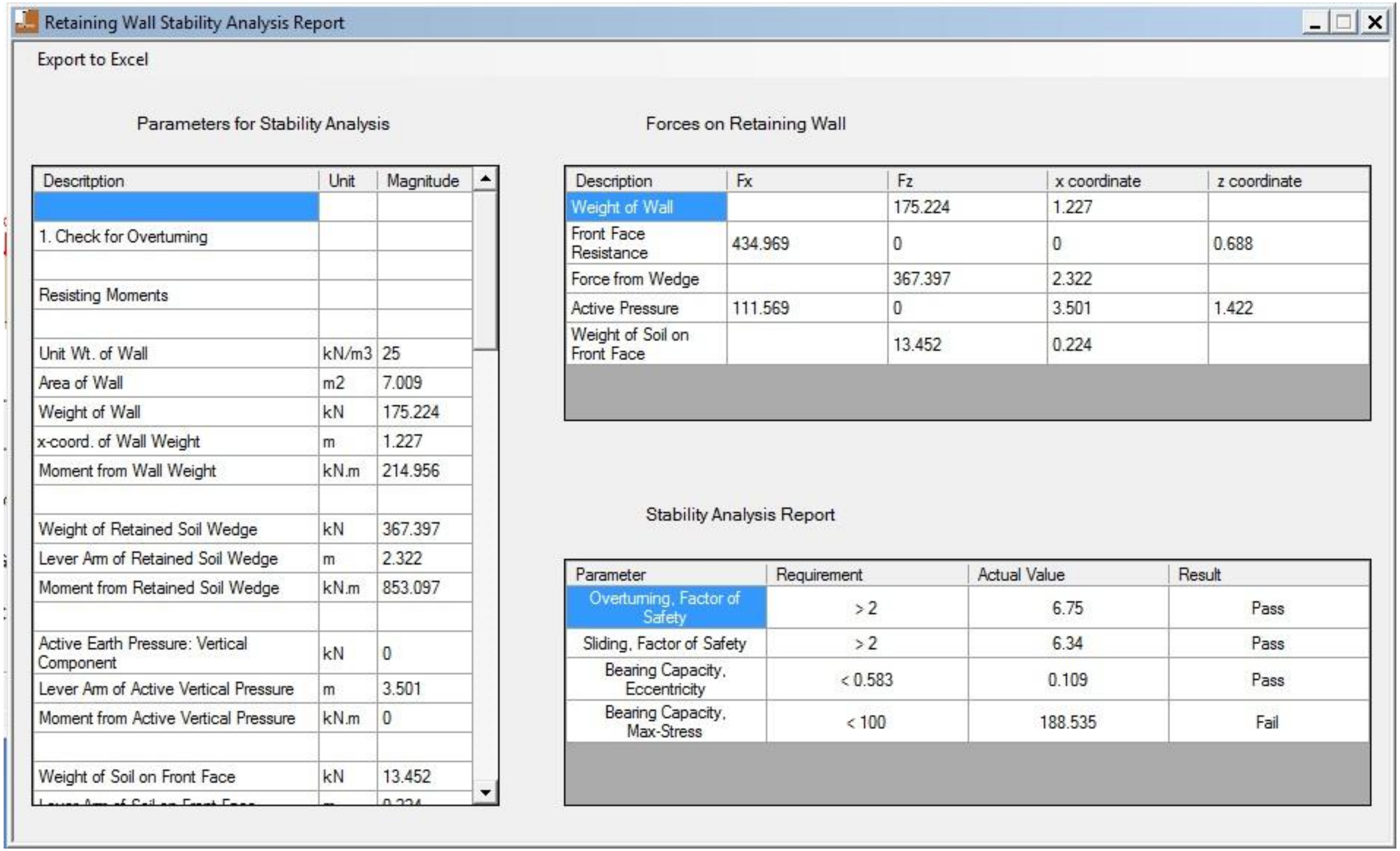


Figure 4. 25: Graphical and tabular output of Coulomb's earth pressure analysis

5. To display the stability calculations, click the “Show Stability Calculations” button. Then the table shown in figure 4.26 shall pop out.



Descritption	Unit	Magnitude
1. Check for Overturning		
Resisting Moments		
Unit Wt. of Wall	kN/m3	25
Area of Wall	m2	7.009
Weight of Wall	kN	175.224
x-coord. of Wall Weight	m	1.227
Moment from Wall Weight	kN.m	214.956
Weight of Retained Soil Wedge	kN	367.397
Lever Arm of Retained Soil Wedge	m	2.322
Moment from Retained Soil Wedge	kN.m	853.097
Active Earth Pressure: Vertical Component	kN	0
Lever Arm of Active Vertical Pressure	m	3.501
Moment from Active Vertical Pressure	kN.m	0
Weight of Soil on Front Face	kN	13.452

Description	Fx	Fz	x coordinate	z coordinate
Weight of Wall		175.224	1.227	
Front Face Resistance	434.969	0	0	0.688
Force from Wedge		367.397	2.322	
Active Pressure	111.569	0	3.501	1.422
Weight of Soil on Front Face		13.452	0.224	

Parameter	Requirement	Actual Value	Result
Overturning, Factor of Safety	> 2	6.75	Pass
Sliding, Factor of Safety	> 2	6.34	Pass
Bearing Capacity, Eccentricity	< 0.583	0.109	Pass
Bearing Capacity, Max-Stress	< 100	188.535	Fail

Figure 4.26: Retaining Wall Stability Analysis Report Table

5. VALIDATION OF THE SOFTWARE

To validate the software developed, examples have been worked out as presented in Annex-C. From the examples, the following observations have been made:

- The results obtained from hand/spread sheet calculations and the software are similar in all analyses.

- The developed software is simpler, faster and more accurate.

6. CONCLUSION AND RECOMMENDATIONS

6.1. Conclusion

So far, the basic steps used in the development of GeoLEPARD software for computing lateral earth pressure and design/analysis of retaining walls have been presented along with comparison of the software's output against hand/spread sheet calculations. Based on the observations made in due course of this research, the following conclusions are drawn.

1) The developed software can be used to analyze lateral earth pressure and design/analyze retaining walls simply and quickly using the earth pressure theories discussed.

2) Usage of commercially available software is limited due to high cost involved in their procurement. Trial versions which are available for free or at lower costs are usually limited in processing capacity. Hence, the software developed herewith provides the opportunity of designing retaining walls at a much convenient and economical manner.

6.2. Recommendation

The application developed here with has taken an initial step in making use of available theories to enable analysis of earth pressure and design/analysis of retaining walls. Either by upgrading this application or developing a similar application with additional features, it is possible to expand the application's use for other geotechnical problems.

Among areas of possible expansion, the following are the major ones.

1) Analysis of Lateral Earth Pressure using elastic theory;

2) Analysis of Lateral Earth Pressure considering earth quake;

3) Analysis/design of retaining walls for their structural (internal) stability;

4) Analysis and design of other type of walls (e.g. counter fort retaining wall, sheet piles, etc)

REFERENCES

[1] Alemayehu Teferra - Principles of Foundation Engineering Addis Ababa University Press, 2008

[2] A.Aysen - Soil Mechanics; Basic Concepts and Engineering Applications. A.A. Balkema, Lisse, The Netherlands, 2002

[3] Braja M. Das - Principles of Foundation Engineering CENGAGE Learning, 2011

[4] Braja M. Das - Theorethical Foundation Engineering J.Ross Publishing, 2007

[5] C.Vencatramaiah - Geotechnical Engineering New Age International Publishers, 2009

[6] European Union - Eurocode-7, Geotechnical design, EN-1997-1 (2004) English

[7] Graham E. Barnes - Soil Mechanics, Principles and Practice Macmilan Publishers Ltd, 2000

[8] John Atkinson - The Mechanics of Soil and Foundations, Taylor & Francis Group, 2007

[9] Joseph E. Bowles - Foundation Analysis and Design McGraw-Hill KoGakusha Ltd, 1977

[10] Joseph E. Bowles - Foundation Analysis and Design McGraw-Hill Companies Inc, 1997

[11] Michael Halvorson - Visual Basic 2010 Step by Step Microsoft Press, 2010

[12] Muni Budhu - Soil Mechanics and Foundations
John Wiley & Sons Inc, 2011

[13] R. F. Craig - Craig's Soil Mechanics
Taylor & Francis Group, 2004

[14] Roy Whitlow - Basic Soil Mechanics
Pearson Education Ltd, 2001

[15] Sam Helwany - Applied Soil Mechanics
John Wiley & Sons Inc, 2007

[16] Samuel Tadesse - Lecture Notes, CENG6208, Ananalysis of Slope, Earth Retaining Structure & Underground Structures

[17] Tim Patrick - ADO.NET 4 Step by Step

[18] V.N.S. Murthy - Geotechnical Engineering, Principles and Practices of Soil Mechanics and Foundation Engineering
Marcel Dekker Inc.

Software Development Environment

- Microsoft Visual Studio 2010 (Visual Basic.net)

ANNEXES

ANNEX – A: - Conditions of Lateral Earth Pressure

A.1 Conditions of at-rest earth pressure

A.1.1 Multilayer soil condition with horizontal ground and vertical wall

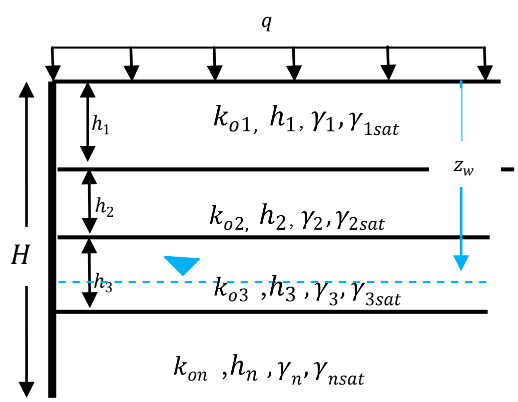


Figure: A-1: Multi layer soils with a retaining wall of vertical backface.

This condition of soil profile is addressed for the cases of; I) with no water, and II) with water.

Where; k_{oi} = at rest earth pressure coefficient of the respective layer

h_i = depth of the respective layer (layer thickness)

z_w = depth of water

γ_i = Bulk unit weight of the respective layer

γ_{isat} = Saturated unit weight of the respective layer

$H = h_1 + h_2 + h_3 + \cdots . + h_n$

q = Uniform Surcharge

I. No water condition ($z_w \geq H$)

$\boldsymbol{\sigma'_h} = k_{o1}(q + \gamma_1 z)$ when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_h} = k_{o2}\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}$ when $h_1 \leq z \leq (h_1 + h_2)$ $[A.1]$

$\boldsymbol{\sigma'_h} = k_{o3}\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}$ when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$

$\boldsymbol{\sigma'_h} = k_{on}\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3\ldots\ldots} + \gamma_n (z - (h_1 + h_2 + h_3 + h_{4+\ldots} h_{n-1}))\}$

when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$

where; $\boldsymbol{\sigma'_h} = lateral\ earth\ pressure(effective\ stress)$

II. Water level above the base (zw < H)

Assuming that the water level is somewhere in the third layer (but it could be anywhere) i.e $z_w \geq (h_1 + h_2)$;

$\boldsymbol{\sigma'_h} = k_{o1}(q + \gamma_1 z)$ when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_h} = k_{o2}\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}$ when $h_1 \leq z \leq h_1 + h_2$

$\boldsymbol{\sigma'_h} = k_{o3}\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}$, when $(h_1 + h_2) \leq z \leq z_w$

$$\boldsymbol{\sigma'_h} = k_{o3}\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\right\} + \gamma_w(z - z_w) \quad [A.2]$$

when $z_w \leq z \leq (h_1 + h_2 + h_3)$

$$\boldsymbol{\sigma'_h} = k_{on}\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((h_1 + h_2 + h_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3) + \ldots + \gamma'_n(z - (h_1 + h_2 + h_3 + \cdots + h_{n-1})\right\} + \gamma_w(z - z_w)$$

when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$

Where $\gamma'_i = (\gamma_{isat} - \gamma_w) = effective\ unit\ weight\ of\ the\ respective\ layer$

$\gamma_w = unit\ weight\ of\ water$

A.1.2 Multilayer soil condition with horizontal ground and inclined wall backface

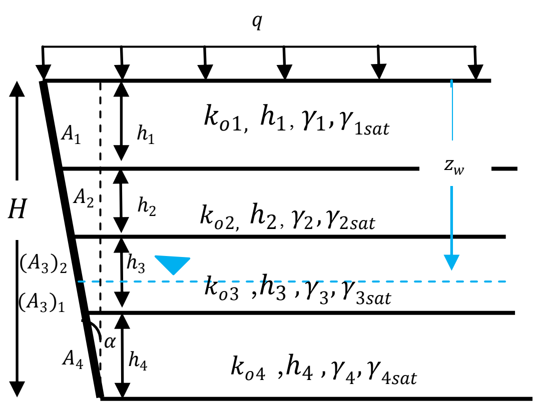


Figure:A-2: Multi layer soils with a retaining wall of inclined back face

To analyze such a case, i.e, inclined wall condition, a fictitious vertical plane is drawn through the heel and the weight of soil of the wedge formed is considered as the vertical component of the lateral earth pressure. The pressure computed at the face of the fictitious plane is considered the horizontal component [4].To come up with the resultant force (R), the vertical and horizontal components are added vectorially.

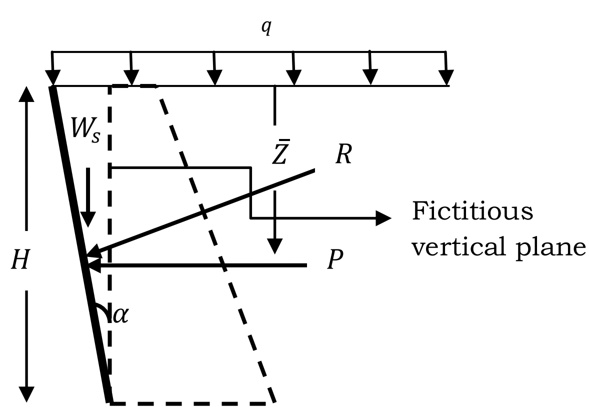


Figure A.3: At-rest force for a retaining wall with an inclined backface.

I. No Water Condition ($z_w \geq H$)

Assuming there are three layers:

Horizontal Component

$\boldsymbol{\sigma'_h} = k_{01}(q + \gamma_1 z)$ when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_h} = k_{02}\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}$ when $h_1 \leq z \leq (h_1 + h_2)$ $[A.3]$

$\boldsymbol{\sigma'_h} = k_{03}\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}$ when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$

Vertical Component

- computing the areas starting from the lower layer (triangle) to the subsequent top layers

Area-3 = $0.5 * (h_3)^2 \tan\alpha$

Area-2 = $0.5 * h_2 * \{2h_3 + h_2\} * tan\alpha$

Area-1 = $0.5 * h_3 * \{h_1 + 2h_2 + 2h_3\} * tan\alpha$ $[A.4]$

= $0.5 * h_3 * \{x + (h_3 + h_2) * tan\alpha\}$

$\boldsymbol{W_s} = \gamma_1 * A_1 + \gamma_2 * A_2 + \gamma_3 * A_3 + q * x$

II. Water level above the base (z_w < H)

Assuming that we have four layers;

Horizontal component

*Assuming that the water level is somewhere in the third layer i.e. $z_w \geq (h_1 + h_2)$;

$\boldsymbol{\sigma'_{ha}} = k_{a1}(q + \gamma_1 z)$ when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_{ha}} = k_{a2}\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}$ when $h_1 \leq z \leq h_1 + h_2$ $[A.5]$

$\boldsymbol{\sigma'_{ha}} = k_{a3}\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}$, when $(h_1 + h_2) \leq z \leq z_w$

$\boldsymbol{\sigma'_{ha}} = k_{a3}\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\right\} + \gamma_w(z - z_w)$

when $z_w \leq z \leq (h_1 + h_2 + h_3)$

$\boldsymbol{\sigma'_{ha}} = k_{a4}\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h}_1 + \mathrm{h}_2 + \mathrm{h}_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3)\} + \gamma_w(z - z_w)$

when $(h_1 + h_2 + h_3) \leq z \leq (h_1 + h_2 + h_3 + h_4)$

Vertical component

To compute the vertical component use Figure 3.2.

$$A_4 = 0.5 * (h_4)^2 \tan\alpha$$

$$(A_3)_2 = 0.5 * [(h_1 + h_2 + h_3) - z_w] * [h_4 \tan\alpha + (\{(h_1 + h_2 + h_3) - z_w\} + h_4) * \tan\alpha]$$

$$(A_3)_1 = 0.5 * [z_w - (h_1 + h_2)] * [(h_3 + h_4) \tan\alpha + (\{(h_1 + h_2 + h_3) - z_w\} + h_4) * \tan\alpha]$$

$$A_2 = 0.5 * h_2 * \{2h_4 + 2h_3 + h_2\} * tan\alpha$$

$$A_1 = 0.5 * h_1 * \{2h_4 + 2h_3 + 2h_2 + h_1\} * tan\alpha \qquad [A.6]$$

$$\boldsymbol{W_s} = \gamma_1 * A_1 + \gamma_2 * A_2 + \gamma_3 * (A_3)_1 + \gamma_{3sat} * (A_3)_2 + \gamma_{4sat} * A_4 + q * x$$

$$R = \sqrt{P^2 + W_s^{\,2}} \qquad [\mathrm{A.7}]$$

A.1.3 Multilayer soil condition with inclined ground and vertical wall backface

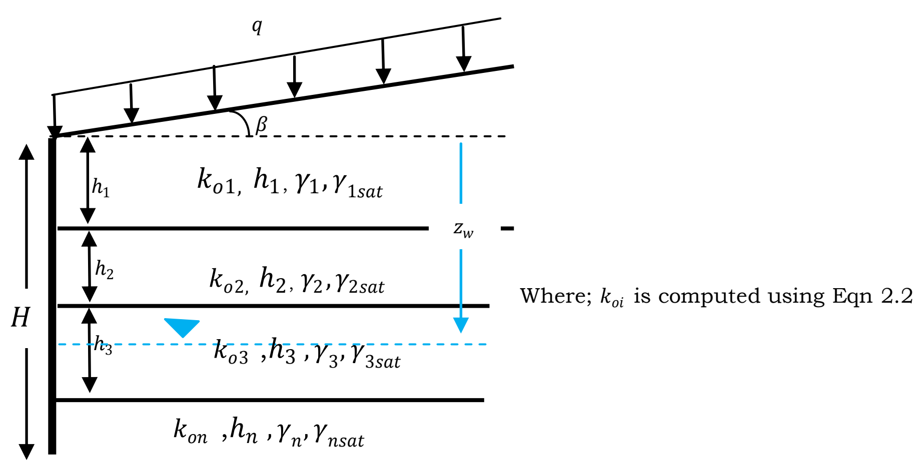


Figure A.4: Multi layer soils with vertical retaining wall and inclined ground

I. No Water Condition ($z_w \geq H$)

$\boldsymbol{\sigma'_h} = k_{01}(1 + sin\beta)(q + \gamma_1 z)$ when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_h} = k_{02}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}$ when $h_1 \leq z \leq (h_1 + h_2)$

$\boldsymbol{\sigma'_h} = k_{03}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\}$

when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$ $[A.8]$

$\boldsymbol{\sigma'_h} = k_{0n}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3\ldots\ldots} + \gamma_n (z - (h_1 + h_2 + h_3 + h_{4+\ldots} h_{n-1}))\}$

when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$

II. Water level above the base ($z_w < H$)

*Assuming that the water level is somewhere in the third layer i.e $z_w \geq (h_1 + h_2)$, but it could be anywhere)

$\boldsymbol{\sigma'_h} = k_{01}(1 + sin\beta)(q + \gamma_1 z)$ when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_h} = k_{02}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}$ when $h_1 \leq z \leq h_1 + h_2$,

$\boldsymbol{\sigma'_h} = k_{03}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\}$, when $(h_1 + h_2) \leq z \leq z_w$ $[A.9]$

$\boldsymbol{\sigma'_h} = k_{03}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\} + \gamma_w(z - z_w)$

when $z_w \leq z \leq (h_1 + h_2 + h_3)$

$$\boldsymbol{\sigma'_h} = k_{04} * (1 + sin\beta)\begin{Bmatrix} q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((h_1 + h_2 + h_3) - z_w) + \\ \gamma'_4(z - (h_1 + h_2 + h_3) \end{Bmatrix} + \gamma_w(z - z_w)$$

when $(h_1 + h_2 + h_3) \leq z \leq (h_1 + h_2 + h_3 + h_4)$

.....

A.1.4 Multilayer soil condition with inclined ground and inclined wall backface

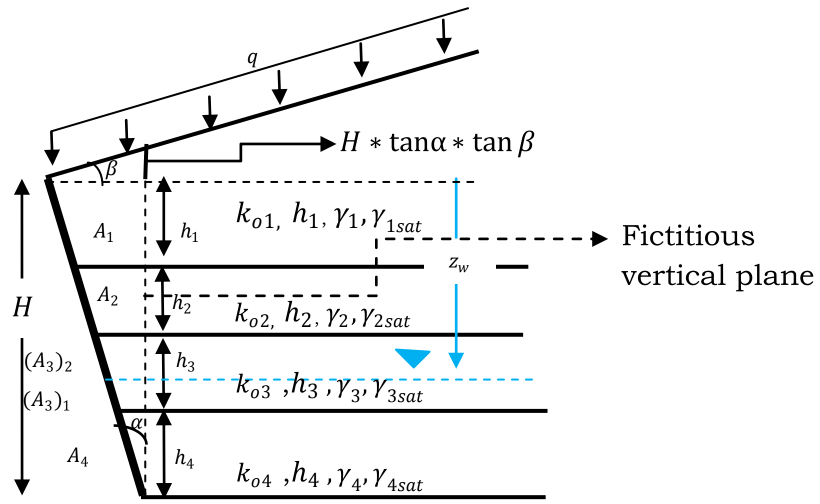


Figure A.5: Multi layer soils with a retaining wall of inclined back face and inclined ground

Just as the case in section A.1.3 above, a fictitious vertical plane is drawn to the ground through the heel and the weight of soil wedge formed is considered as part the vertical component of the lateral earth pressure[4]. Due to the inclination of both the wall and ground both the horizontal and vertical components introduce an additional height which is equal to $H * \tan\alpha * \tan\beta$.

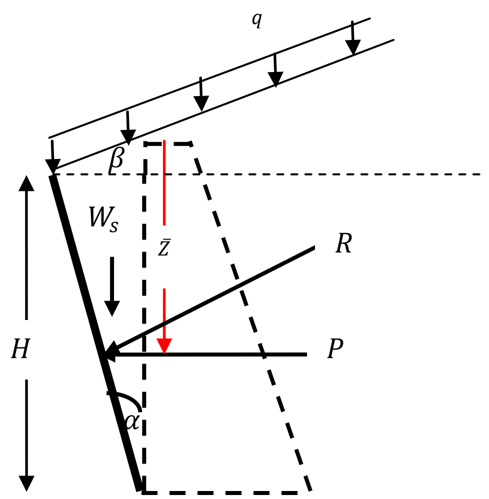


Figure A.6: At-rest force for a retaining wall with an inclined backface and inclined ground.

I. No Water Condition ($z_w \geq H$)

Assuming there are three layers;

<u>Horizontal Component</u>

$\boldsymbol{\sigma'_h} = k_{01}(1 + sin\beta)(q + \gamma_1 z)$ when $0 \leq z \leq (h_1 + H * \tan\alpha * \tan\beta)$

$\boldsymbol{\sigma'_h} = k_{02}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}$ when $(h_1 + H * \tan\alpha * \tan\beta) \leq z \leq (h_1 + h_2 + H * \tan\alpha * \tan\beta)$

$\boldsymbol{\sigma'_h} = k_{03}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}$ when $(h_1 + h_2 + H * \tan\alpha * \tan\beta)$ $[A.10]$

$\leq z \leq (h_1 + h_2 + h_3 + H * \tan\alpha * \tan\beta)$

<u>Vertical Component</u>

- computing the areas starting from the lower layer (triangle) to the subsequent top layers, i.e, up to the smaller top most wedge;

$A_3 = 0.5 * (h_3)^2 \tan\alpha$

$A_2 = 0.5 * h_2 * \{2h_3 + h_2\} * tan\alpha$ $[A.11]$

$A_1 = 0.5 * h_3 * \{h_1 + 2h_2 + 2h_3\} * tan\alpha$

$= 0.5 * h_3 * \{x + (h_3 + h_2) * tan\alpha\}$

$A_{wedge} = \frac{1}{2}x^2 tan\beta$

$\boldsymbol{W_s} = \gamma_1 * A_1 + \gamma_2 * A_2 + \gamma_3 * A_3 + \gamma_3 * A_{wedge} + q * \frac{H * tan\alpha}{\cos\beta}$

II. Water level above the base (zw < H)

Assuming that we have four layers and the water level lies somewhere in the third layer;

Horizontal component

*Assuming that the water level is somewhere in the third layer i.e. $z_w \geq (h_1 + h_2)$;

$\boldsymbol{\sigma'_h} = k_{o1}(1 + sin\beta)(q + \gamma_1 z) \qquad when\ 0 \leq z \leq (h_1 + H * tan\alpha * tan\,\beta)$

$\boldsymbol{\sigma'_h} = k_{02}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \quad$ when $(h_1 + H * \tan\alpha * \tan\beta) \leq z \leq (h_1 + h_2 + H * \tan\alpha * \tan\beta)$ $[A.12]$

$when\ (h_1 + H * tan\alpha * tan\,\beta) \leq z \leq (h_1 + h_2 + H * tan\alpha * tan\,\beta)$

$\boldsymbol{\sigma'_h} = k_{03}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}$,

$when\ (h_1 + h_2 + H * tan\alpha * tan\,\beta) \leq z \leq (z_w + H * tan\alpha * tan\,\beta)$

$\boldsymbol{\sigma'_h} = k_{03}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\} + \gamma_w(z - z_w)$

$when\ (z_w + H * tan\alpha * tan\,\beta) \leq z \leq (h_1 + h_2 + h_3 + H * tan\alpha * tan\,\beta)$

$$\boldsymbol{\sigma'_h} = k_{o4}(1 + sin\beta)\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((h_1 + h_2 + h_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3)\} + \gamma_w(z - z_w)$$

$when\ (h_1 + h_2 + h_3 + H * tan\alpha * tan\,\beta) \leq z \leq (h_1 + h_2 + h_3 + h_4 + H * tan\alpha * tan\,\beta)$

Vertical component

We compute the areas starting from the last layer;

$A_4 \quad = 0.5 * (h_4)^2 \tan\alpha$

$(A_3)_2 = 0.5 * [(h_1 + h_2 + h_3) - z_w] * [h_4 \tan\alpha + (\{(h_1 + h_2 + h_3) - z_w\} + h_4) * \tan\alpha]$

$(A_3)_1 = 0.5 * [z_w - (h_1 + h_2)] * [(h_3 + h_4)\tan\alpha + (\{(h_1 + h_2 + h_3) - z_w\} + h_4) * \tan\alpha]$

$A_2 = 0.5 * h_2 * \{2h_4 + 2h_3 + h_2\} * tan\alpha$ $[A.13]$

$A_1 = 0.5 * h_1 * \{2h_4 + 2h_3 + 2h_2 + h_1\} * tan\alpha$

$A_{wedge} = \frac{1}{2} * (H * tan\alpha)^2 * tan\beta$

$\boldsymbol{W_s} = A_{wedge} * \gamma_1 + \gamma_1 * A_1 + \gamma_2 * A_2 + \gamma_3 * (A_3)_1 + \gamma_{3sat} * (A_3)_2 + \gamma_{4sat} * A_4 + q * \frac{H * tan\alpha}{\cos\beta}$

Conditions of Rankine(Bell) earth pressure theory

A.2.1 Friction Soils

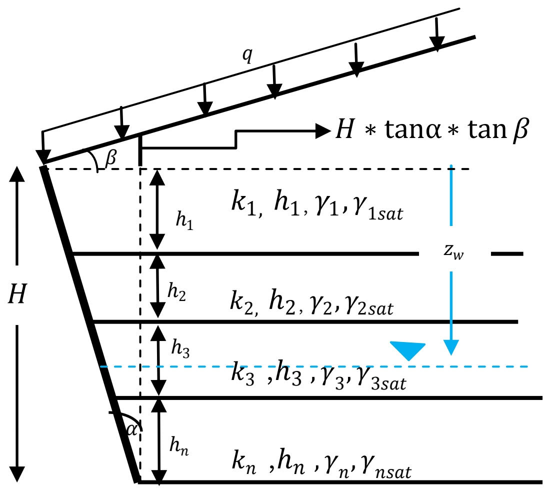


Figure A.7: Multi layer soils with a retaining wall of inclined back face and inclined ground

Where; k_i = Active/passive Rankine earth pressure coefficient of the respective layer given by equations Eqn 2.26 and 2.27 respectively

As Rankine's theory assumes that the retaining wall is smooth and vertical. Hence, a fictitious vertical plane is drawn through the heel to the ground and the pressure is computed at the face of the fictitious plane.

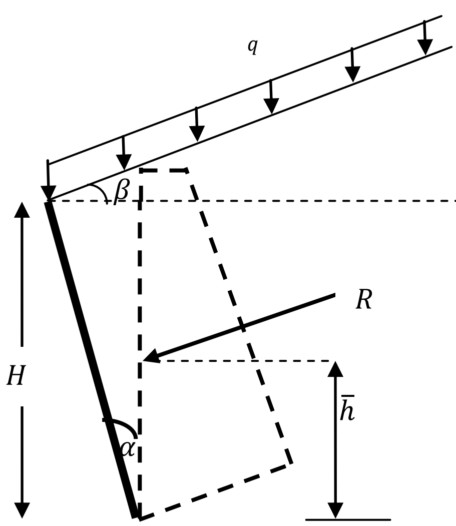


Figure A.8: Lateral earth pressure force due to Rankine's theory for a retaining wall with an inclined back face and inclined ground.

I. No Water Condition ($z_w \geq H$)

$\sigma'_h = k_1(q + \gamma_1 z)\cos\beta \quad H.C$

$\sigma'_h = k_1(q + \gamma_1 z)\sin\beta \quad V.C$

when $0 \leq z \leq h_1$

$\sigma'_h = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\cos\beta \quad H.C$

$\sigma'_h = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\sin\beta \quad V.C$

when $h_1 \leq z \leq (h_1 + h_2)$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}\cos\beta \quad H.C$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}\sin\beta \quad V.C$ $\quad [A.14]$

when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$

$\sigma'_h = k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3\ldots\ldots} + \gamma_n\,(z - (h_1 + h_2 + h_3 + h_{4+\ldots}h_{n-1}))\}\cos\beta \quad H.C$

$\sigma'_h = k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3\ldots\ldots} + \gamma_n\,(z - (h_1 + h_2 + h_3 + h_{4+\ldots}h_{n-1}))\}\sin\beta \quad V.C$

when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$

II. Water level above the base (zw < H)

*Assuming that the water level is somewhere in the third layer i.e $z_w \geq (h_1 + h_2)$;

$\sigma'_h = k_1(q + \gamma_1 z)\cos\beta \quad H.C$

$\sigma'_h = k_1(q + \gamma_1 z)\sin\beta \quad V.C$

when $0 \leq z \leq h_1$

$\sigma'_h = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\cos\beta \quad H.C$

$\sigma'_h = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\sin\beta \quad V.C$

when $h_1 \leq z \leq h_1 + h_2$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\}\cos\beta\ , H.C$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\}\sin\beta, \quad V.C$ $\quad [A.15]$

when $(h_1 + h_2) \leq z \leq z_w$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\}\cos\beta + \gamma_w(z - z_w) \quad H.C$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\}\sin\beta \quad V.C$

when $z_w \leq z \leq (h_1 + h_2 + h_3)$

$$\boldsymbol{\sigma'_h} = k_4\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h}_1 + \mathrm{h}_2 + \mathrm{h}_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3))\}\cos\beta + \gamma_w(z - z_w) \qquad H.C$$

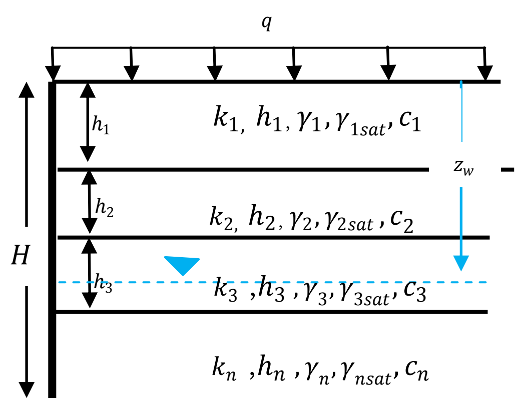
$$\boldsymbol{\sigma'_h} = k_4\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h}_1 + \mathrm{h}_2 + \mathrm{h}_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3))\}\sin\beta \qquad V.C$$

when $(h_1 + h_2 + h_3) \leq z \leq (h_1 + h_2 + h_3 + h_4)$

.....

Note: The hydrostatic pressure is added only to the horizontal component.

A.2.2 Mixed Soils



where; c_i = Cohesion for the respective layer

$k_i = Active/passive\ earth\ pressure\ coefficients\ given\ by\ Eqn. 2.19a\ and\ 2.19b\ respectively$

Figure A.9: Multi layered mixed soils with a retaining wall of vertical backface and horizontal ground.

I. No Water Condition ($z_w \geq H$)

$\sigma'_h k_1(q + \gamma_1 z) \pm 2c_1\sqrt{k_1}$ *when* $0 \leq z \leq h_1$

$\sigma'_h = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm 2c_2\sqrt{k_2}$ *when* $h_1 \leq z \leq (h_1 + h_2)$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\} \pm 2c_3\sqrt{k_3}$ $[A.16]$

when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$

$\sigma'_h = k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3......} + \gamma_n (z - (h_1 + h_2 + h_3 + h_{4+...} h_{n-1}))\} \pm 2c_n\sqrt{k_n}$

when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$

II. Water level above the base ($z_w < H$)

*Assuming that the water level is somewhere in the third layer i.e $z_w \geq (h_1 + h_2)$;

$\sigma'_h = k_1(q + \gamma_1 z) \pm 2c_1\sqrt{k_1}$ *when* $0 \leq z \leq h_1$

$\sigma'_h = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm 2c_2\sqrt{k_2}$ *when* $h_1 \leq z \leq h_1 + h_2$ $[A.17]$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\} \pm 2c_3\sqrt{k_3}$, *when* $(h_1 + h_2) \leq z \leq z_w$

$\sigma'_h = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\} \pm 2c_3\sqrt{k_3} + \gamma_w(z - z_w)$
when $z_w \leq z \leq (h_1 + h_2 + h_3)$

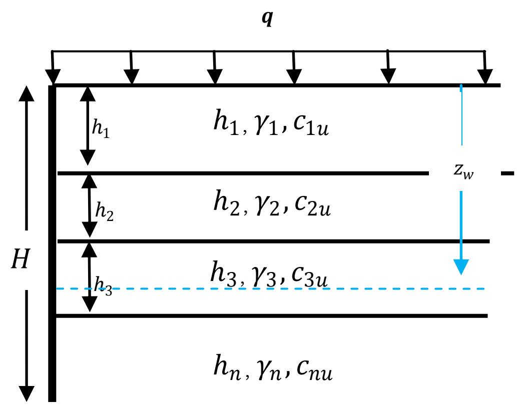
$$\sigma'_h = k_4\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h_1} + \mathrm{h_2} + \mathrm{h_3}) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3))\} \pm 2c_4\sqrt{k_4} + \gamma_w(z - z_w)$$

when $(h_1 + h_2 + h_3) \leq z \leq (h_1 + h_2 + h_3 + h_4)$

.....

Note: In the above equations at the (±) part, the (+) sign shall be considered for passive case and the (−) sign for active case.

A.2.3 Layered soils in case of Total Stress



where; c_{iu} = Undrained cohesion for the respective layer

Figure A.10: Multi layered soils in undrained condition

A.2.3.1 Conditions for Active and Passive pressures for Total Stress Case

$\sigma_h = \{q + \gamma_1 z\} \pm 2c_{1u}$ *when* $0 \le z \le h_1$

$\sigma_h = \{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm 2c_{2u}$ *when* $h_1 \le z \le h_1 + h_2$ $[A.18]$

$\sigma_h = \{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\} \pm 2c_{3u}$ *when* $h_1 + h_2 \le z \le h_1 + h_2 + h_3$

$\sigma_h = \{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(h_1 + h_2 + h_3) + \cdots + \gamma_n (z - (h_1 + h_2 + h_3 + h_{4+\dots} h_{n-1}))\} \pm 2c_{nu}$

when $(h_1 + h_2 + h_3 + \dots + h_{n-1}) \le z \le (h_1 + h_2 + h_3 + h_4 + \dots h_n)$

Note:

- Total stress case uses just the saturated (undrained) unit weight of soil. The depth of water level shall not be put in to consideration. Irrespective of the water level the computation for lateral pressure goes the same.
- In the above equations at the ($\pm$) part, the (+) sign shall be considered for the passive case and the ($-$) sign for active case.

A.3 Conditions of Coulomb's earth pressure theory

A.3.1 Friction Soils

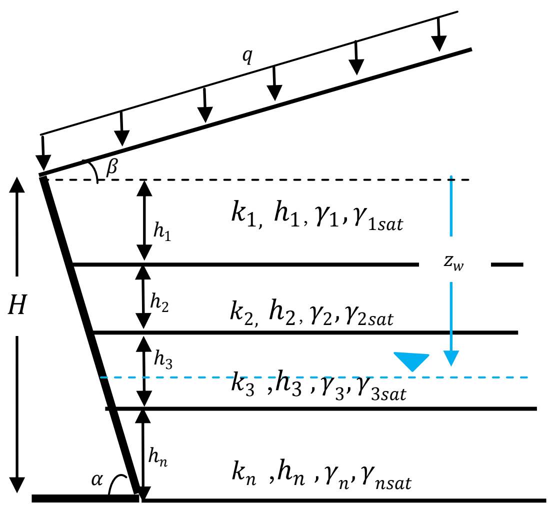
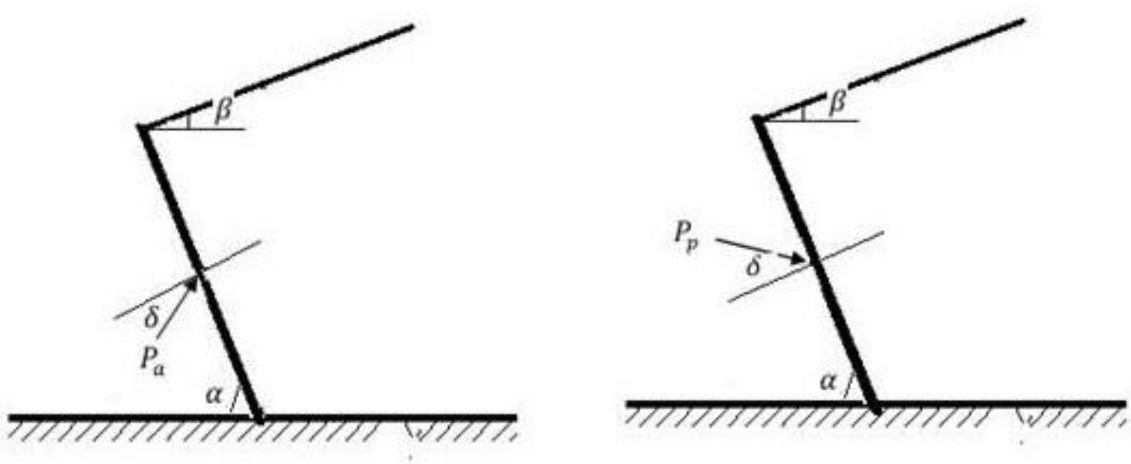


Figure A.11: Multi layer soils with a retaining wall of inclined back face and inclined ground.

Where; k_i = Active/passive Coulomb's earth pressure coefficient of the respective layer given by equations Eqn: 2.44 and 2.56 respectively

As coulomb's theory assumes wall friction, the resultant force (P) makes an angle (δ) with the normal to the back face of the wall.



(a) Active case (b) Passive case

Figure A.12: Lateral earth pressure force due to Coulomb's theory;

I. **No Water Condition ($z_w \geq H$)**

$\boldsymbol{\sigma'_h} = k_1(q + \gamma_1 z)\cos\big((\delta_1 - \alpha) + 90\big) \quad H.C$

when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_h} = k_1(q + \gamma_1 z)\sin\big((\delta_1 - \alpha) + 90\big) \quad V.C$

$\boldsymbol{\sigma'_h} = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\cos\big((\delta_2 - \alpha) + 90\big) \quad H.C$

when $h_1 \leq z \leq (h_1 + h_2)$

$\boldsymbol{\sigma'_h} = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\sin\big((\delta_2 - \alpha) + 90\big) \quad V.C$

$\boldsymbol{\sigma'_h} = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}\cos\big((\delta_3 - \alpha) + 90\big) \quad H.C$ $\quad [A.19]$

$\boldsymbol{\sigma'_h} = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\}\sin\big((\delta_3 - \alpha) + 90\big) \quad V.C$

when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$

$\boldsymbol{\sigma'_h} = k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3\, h_{3 \ldots\ldots} + \gamma_n\,(z - (h_1 + h_2 + h_3 + h_{4\,+\ldots} h_{n-1}))\}\cos\big((\delta_n - \alpha) + 90\big) \quad H.C$

$\boldsymbol{\sigma'_h} = k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3\, h_{3 \ldots\ldots} + \gamma_n\,(z - (h_1 + h_2 + h_3 + h_{4\,+\ldots} h_{n-1}))\}\sin\big((\delta_n - \alpha) + 90\big) \quad V.C$ *when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$*

II. **Water level above the base ($z_w < H$)**

*Assuming that the water level is somewhere in the third layer i.e $z_w \geq (h_1 + h_2)$;

$\boldsymbol{\sigma'_h} = k_1(q + \gamma_1 z)\cos\big((\delta_1 - \alpha) + 90\big) \quad H.C$

when $0 \leq z \leq h_1$

$\boldsymbol{\sigma'_h} = k_1(q + \gamma_1 z)\sin\big((\delta_1 - \alpha) + 90\big) \quad V.C$

$\boldsymbol{\sigma'_h} = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\cos\big((\delta_2 - \alpha) + 90\big) \quad H.C$

when $h_1 \leq z \leq h_1 + h_2$

$\boldsymbol{\sigma'_h} = k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\}\sin\big((\delta_2 - \alpha) + 90\big) \quad V.C$

$\boldsymbol{\sigma'_h} = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\}\cos\big((\delta_3 - \alpha) + 90\big) \quad H.C$ $\quad [A.20]$

$\boldsymbol{\sigma'_h} = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\}\sin\big((\delta_3 - \alpha) + 90\big) \quad V.C$

when $(h_1 + h_2) \leq z \leq z_w$

$\boldsymbol{\sigma'_h} = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\}\cos((\delta_3 - \alpha) + 90) + \boldsymbol{\gamma_w(z - z_w)} \quad H.C$

$\boldsymbol{\sigma'_h} = k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\}\sin\beta \quad V.C$

when $z_w \leq z \leq (h_1 + h_2 + h_3)$

$\boldsymbol{\sigma'_h} = k_4\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h}_1 + \mathrm{h}_2 + \mathrm{h}_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3))\}\cos((\delta_4 - \alpha) + 90) + \gamma_w(z - z_w)$
$H.C$

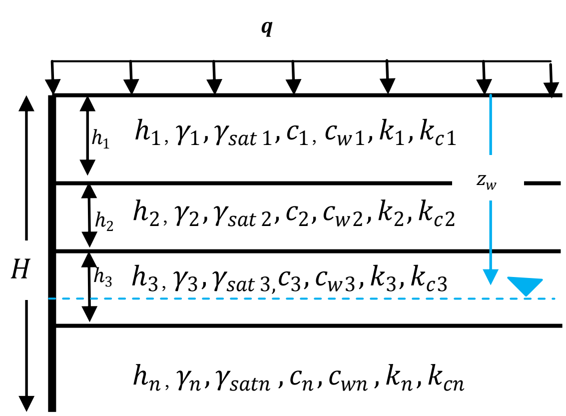
$\boldsymbol{\sigma'_h} = k_4\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h}_1 + \mathrm{h}_2 + \mathrm{h}_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3))\}\sin((\delta_4 - \alpha) + 90)$
$V.C$

when $(h_1 + h_2 + h_3) \leq z \leq (h_1 + h_2 + h_3 + h_4)$

.....

NB: For passive cases the resolving angle shall be; $(\delta + \alpha) - 90$

A.3.2 Mixed Soils



where; c_{wi} = Soil-structure adhesion for the respective layer

k_i = Active/passive earth pressure for the respective layer (Eqns; 2.44 & 2.56) respectively.

$k_{ci} = 2 * \sqrt{k_i(1 + \frac{c_{wi}}{c_i})}$

Figure A.13: Multilayered soils for the analysis of coulomb's condition

I. No Water Condition ($z_w \geq H$)

$$\sigma'_h = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \cos\delta_1 \quad \text{H.C}$$
when $0 \leq z \leq h_1$
$$\sigma'_h = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \sin\delta_1 \quad \text{V.C}$$

$$\sigma'_h = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{c2}] \cos\delta_2 \quad \text{H.C}$$
when $h_1 \leq z \leq (h_1 + h_2)$
$$\sigma'_h = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{ac2}] \sin\delta_2 \quad \text{V.C}$$

$$\sigma'_h = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\} \pm c'_3 k_{c3}] \cos\delta_3 \quad \text{H.C}$$
$$[A.21]$$
$$\sigma'_h = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\} \pm c'_3 k_{c3}\ ] \sin\delta_3 \quad \text{V.C}$$

when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$

$$\sigma'_h = [k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3\, h_{3\ldots\ldots} + \gamma_n\,(z - (h_1 + h_2 + h_3 + h_{4\,+\ldots} h_{n-1}))\} \pm c'_n k_{cn}] \cos\delta_n \quad \text{H.C}$$

$$\sigma'_h = [k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3\, h_{3\ldots\ldots} + \gamma_n\,(z - (h_1 + h_2 + h_3 + h_{4\,+\ldots} h_{n-1}))\} \pm c'_n k_{cn}] \sin\delta_n \quad \text{V.C}$$

when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$

II. Water level above the base ($z_w < H$)

*Assuming that the water level is somewhere in the third layer i.e $z_w \geq (h_1 + h_2)$, but it could be anywhere)

$$\sigma'_h = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \cos\delta_1 \quad H.C$$
when $0 \leq z \leq h_1$
$$\sigma'_h = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \sin\delta_1 \quad V.C$$

$$\sigma'_h = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{c2}] \cos\delta_2 \quad H.C$$
when $h_1 \leq z \leq (h_1 + h_2)$
$$\sigma'_h = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{c2}] \sin\delta_2 \quad V.C$$

$$\sigma'_h = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\} \pm c'_3 k_{c3}] \cos\delta_3\ , H.C$$
$$[A.22]$$
$$\sigma'_h = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\} \pm c'_3 k_{c3}] \sin\delta_3\ , V.C$$

when $(h_1 + h_2) \leq z \leq z_w$

$$\sigma'_h = \left[k_3\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\right\} \pm c'_3 k_{c3}\right] \cos\delta_3 + \gamma_w(z - z_w) \quad H.C$$

$$\sigma'_h = \left[k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 (z_w - (h_1 + h_2)) + \gamma'_3 (z - z_w)\} \pm c'_3 k_{c3}\right] \sin\delta_3 \quad V.C$$

when $z_w \leq z \leq (h_1 + h_2 + h_3)$

$$\sigma'_h = \left[k_4\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 (z_w - (h_1 + h_2)) + \gamma'_3 ((h_1 + h_2 + h_3) - z_w) + \gamma'_4 (z - (h_1 + h_2 + h_3)\} \pm c'_4 k_{c4}\right] \cos\delta_4 + \gamma_w (z - z_w)$$

$H.C$

$$\sigma'_h = \left[k_4\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 (z_w - (h_1 + h_2)) + \gamma'_3 ((h_1 + h_2 + h_3) - z_w) + \gamma'_4 (z - (h_1 + h_2 + h_3)\} \pm c'_4 k_{c4}\right] \sin\delta_4$$

$V.C$

when $(h_1 + h_2 + h_3) \leq z \leq (h_1 + h_2 + h_3 + h_4)$

.....

A.3.3 Layered soils in case of Total Stress ($\emptyset_u = 0$ and $\tau_f = c_u$)

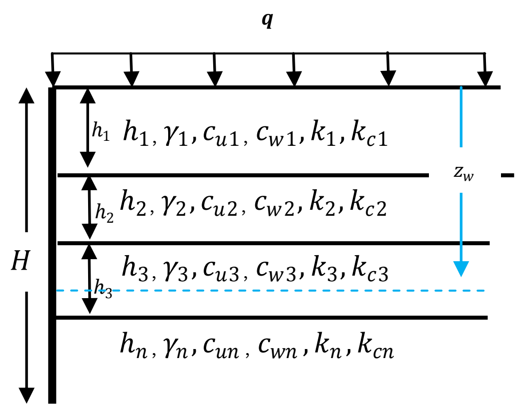


Figure A.14: Multi layered soils in undrained coulomb's condition

For this case, i.e. $\emptyset = 0,\ \beta = 0\ and\ \alpha = 90^0$

$k_i = 1$, both for active and passive conditions

$$Hence;\ k_{ci} = 2 * \sqrt{(1 + \frac{c_{wi}}{c_i})}$$

Moreover, there shall be no vertical component of the lateral earth pressure.

A.3.3.1 Conditions for Active and Passive pressures for Total Stress Case

$$\sigma_h = (q + \gamma_1 z) \pm c_{u1}k_{c1} \quad when\ 0 \le z \le h_1$$

$$\sigma_h = \{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c_{u2}k_{c2} \quad when\ h_1 \le z \le (h_1 + h_2) \qquad [A.23]$$

$$\sigma_h = \{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\} \pm c_{u3}k_{c3} \quad when\ (h_1 + h_2) \le z \le (h_1 + h_2 + h_3)$$

$$\sigma_h = \{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3\ldots\ldots} + \gamma_n (z - (h_1 + h_2 + h_3 + h_{4+\ldots} h_{n-1}))\} \pm c_{un} k_{cn}$$

$$when\ (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \le z \le (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$$

A.4 Conditions of Kerisel and Absi’s earth pressure theory

This theory applies only to horizontal ground ,vertical wall and effective stress case.

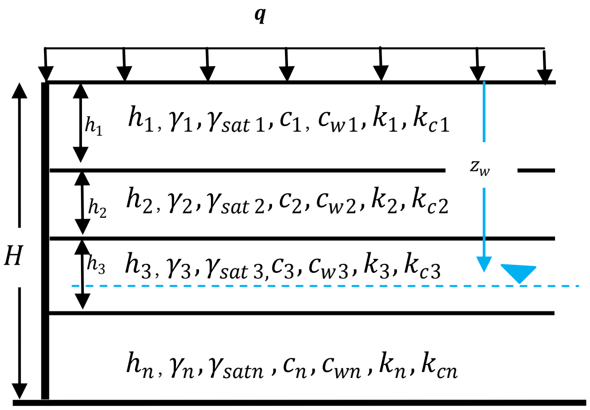
$\sigma_h = K_i \sigma'_v \pm K_{ci} c'$ = Active/passive horizontal component for vertical wall

Where; K_i = active/passive horizontal pressure coefficient.

They are read from the chart given by Kerisel and Absi (Fig; 2.21)

Values of K_{ci} can be obtained with sufficient accuracy from Eqn 2.66 and 2.67:

$$K_{ci} = 2\sqrt{K_i\left(1 + {c'_w}/{c'}\right)}$$



$where;\ c_{wi}$ = Soil-structure adhesion for the respective layer

k_i = Active/passive earth pressure coeficient

Figure A.15: Multi layered soils for the analysis of Kerisel & Absi condition

I. No Water Condition ($z_w \geq H$)

$$\boldsymbol{\sigma_h} = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \quad H.C$$

$$\boldsymbol{\sigma_h} = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \tan\delta_1 \quad V.C$$

when $0 \leq z \leq h_1$

$$\boldsymbol{\sigma_h} = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{c2}] \quad H.C$$

$$\boldsymbol{\sigma_h} = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{c2}] \tan\delta_2 \quad V.C \qquad [A.24]$$

when $h_1 \leq z \leq (h_1 + h_2)$

$$\boldsymbol{\sigma_h} = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\} \pm c'_3 k_{c3}] \quad H.C$$

$$\boldsymbol{\sigma_h} = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2)\} \pm c'_3 k_{c3}] \tan\delta_3 \quad V.C$$

when $(h_1 + h_2) \leq z \leq (h_1 + h_2 + h_3)$

$$\boldsymbol{\sigma_h} = [k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3......} + \gamma_n (z - (h_1 + h_2 + h_3 + h_{4+...}h_{n-1}))\} \pm c'_n k_{cn}] \quad H.C$$

$$\boldsymbol{\sigma_h} = [k_n\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3 h_{3......} + \gamma_n (z - (h_1 + h_2 + h_3 + h_{4+...}h_{n-1}))\} - c'_n k_{cn}] \tan\delta_n \quad V.C$$

when $(h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_{n-1}) \leq z \leq (h_1 + h_2 + h_3 + h_4 + \ldots\ldots h_n)$

II. Water level above the base (zw < H)

*Assuming that the water level is somewhere in the third layer i.e $z_w \geq (h_1 + h_2)$;

$$\sigma_h = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \quad H.C$$

$$\sigma_h = [k_1\{q + \gamma_1 z\} \pm c'_1 k_{c1}] \tan \delta_1 \quad V.C \qquad when\ 0 \leq z \leq h_1 \qquad [A.25]$$

$$\sigma_h = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{c2}] \quad H.C$$

$$\sigma_h = [k_2\{q + \gamma_1 h_1 + \gamma_2(z - h_1)\} \pm c'_2 k_{c2}] \tan \delta_2 \quad V.C$$

when $h_1 \leq z \leq (h_1 + h_2)$

$$\sigma_h = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\} \pm c'_3 k_{c3}] \quad H.C$$

$$\sigma_h = [k_3\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z - (h_1 + h_2))\} \pm c'_3 k_{c3}] \tan \delta_3 \quad V.C$$

when $(h_1 + h_2) \leq z \leq z_w$

$$\sigma_h = \left[k_3\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\right\} \pm c'_3 k_{c3}\right] + \gamma_w(z - z_w) \quad H.C$$

$$\sigma_h = \left[k_3\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3(z - z_w)\right\} \pm c'_3 k_{c3}\right] \tan \delta_3 \quad V.C$$

when $z_w \leq z \leq (h_1 + h_2 + h_3)$

$$\sigma_h = \left[k_4\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h}_1 + \mathrm{h}_2 + \mathrm{h}_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3)\right\} \pm c'_4 k_{c4}\right] + \gamma_w(z - z_w) \quad H.C$$

$$\sigma_h = \left[k_4\left\{q + \gamma_1 h_1 + \gamma_2 h_2 + \gamma_3(z_w - (h_1 + h_2)) + \gamma'_3((\mathrm{h}_1 + \mathrm{h}_2 + \mathrm{h}_3) - z_w) + \gamma'_4(z - (h_1 + h_2 + h_3)\right\} \pm c'_4 k_{c4}\right] \tan \delta_3 \quad V.C$$

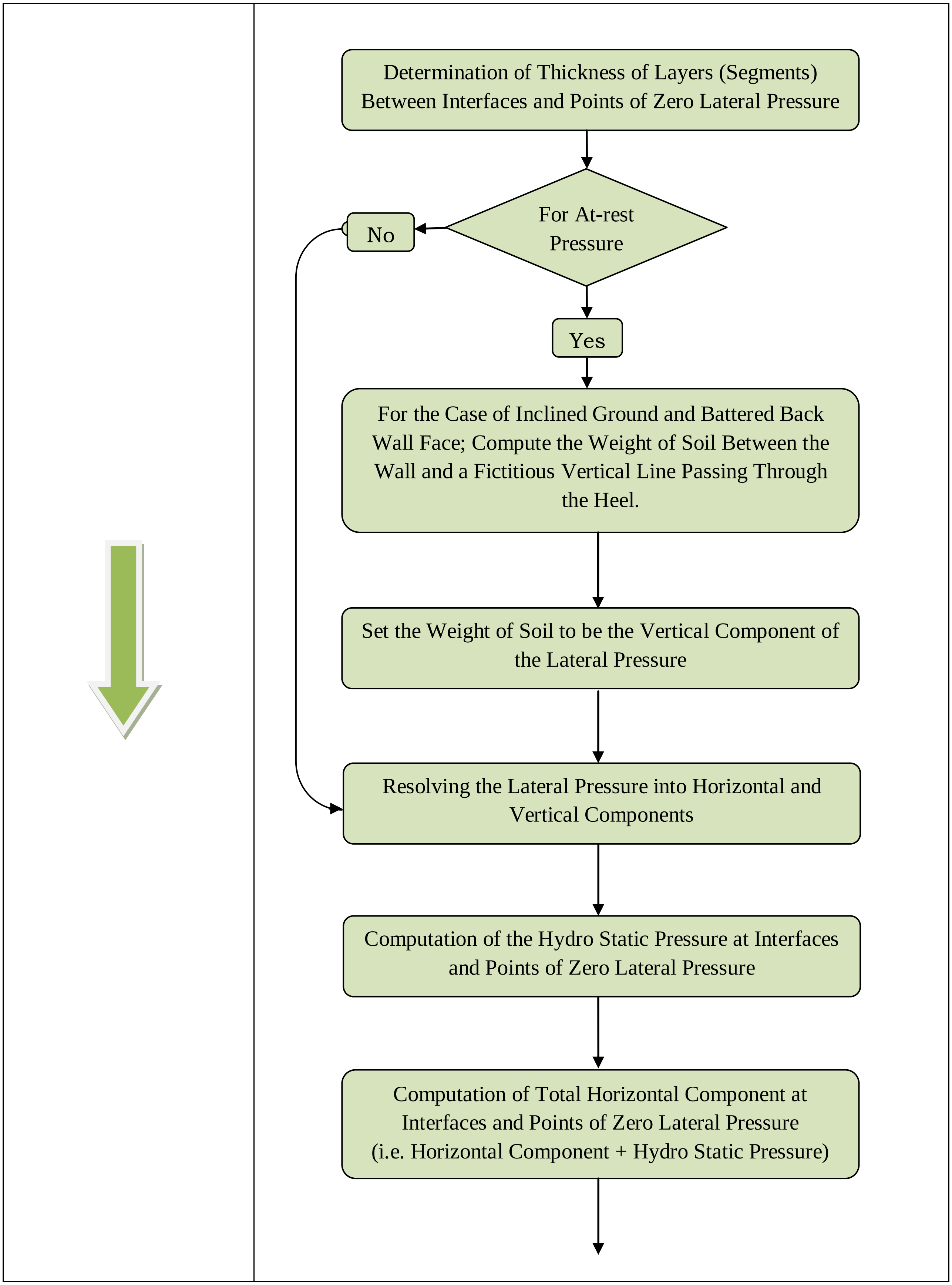
when $(h_1 + h_2 + h_3) \leq z \leq (h_1 + h_2 + h_3 + h_4)$ ....

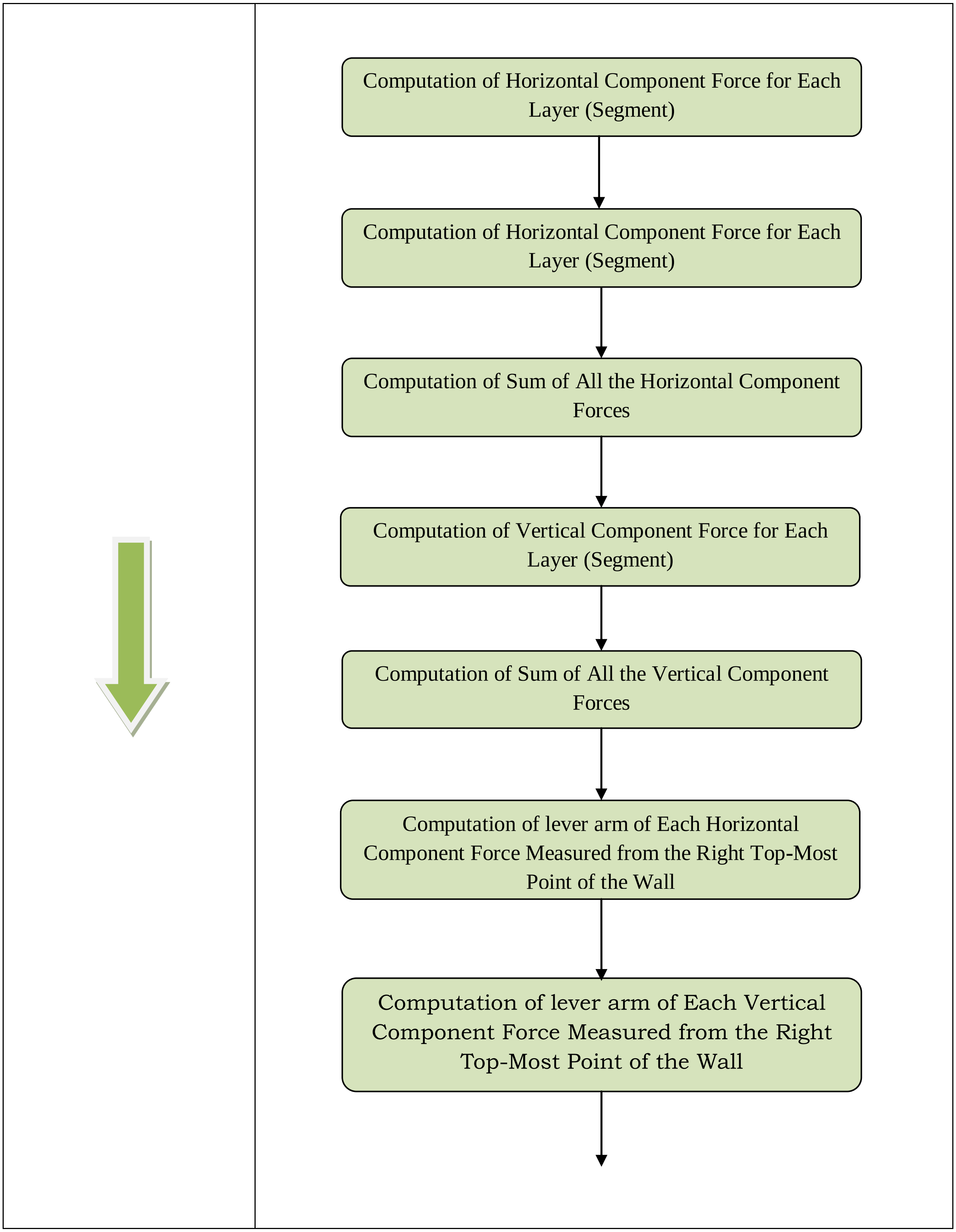
ANNEX –B: - Flow Chart

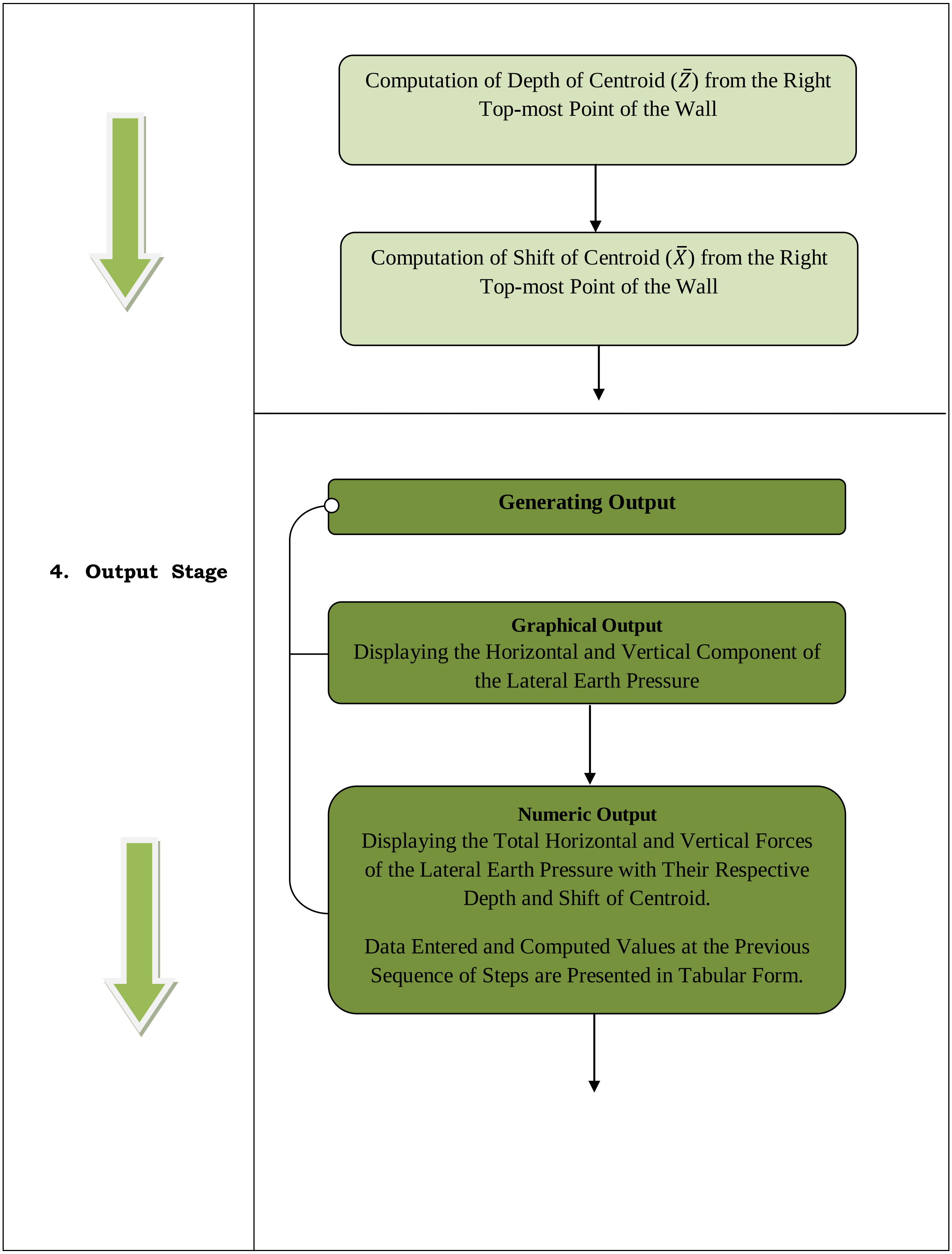
I. **Lateral Earth Pressure**	
1. **Theory Selection Stage**	Selection of Type of Earth Pressure (At-rest, Active, Passive) & Lateral Earth Pressure Theory (Rankine / Coulomb / Kerisel & Absi).
2. **Data Entry Stage**	Inputting Angle of Ground Inclination, Angle of Back Wall Face Inclination, Wall Height, Depth and Thickness of Layers, Soil Data ↓ Entering Soil Data; Bulk Unit Weight of Soil, Stress State, Angle of Internal Friction, Structure-Soil Friction, Cohesion, Adhesion, Parameters for At-rest Pressure, Saturated Unit Weight of Soil, Unit Weight of Water ↓ Assigning Soil to Each Layer ↓ Entering Depth of Ground Water Level, Magnitude of Uniform Surcharge ↓

3. Analysis Stage

1. Calculation of At-rest/Active/Passive Earth Pressure Coefficient for Each Defined Soil
2. Identify Level (Depth) of Each Interface
3. Identify Level (Depth) of Ground Water
4. Computation of Cohesion Component of Earth Pressure
5. Computation of Vertical Pressure at Each Interface
6. Computation of Lateral Pressure at Each Interface
7. Identification of Points (levels) Where the Lateral Pressure is Zero or Negative.
8. Adjustment of Lateral Pressure is for Negative values (i.e. Set Negative Values to Zero).
9. Identification Depths (Levels) Where the Adjusted Pressure is Zero


Determination of Thickness of Layers (Segments) Between Interfaces and Points of Zero Lateral Pressure
For At-rest Pressure
No
Yes
For the Case of Inclined Ground and Battered Back Wall Face; Compute the Weight of Soil Between the Wall and a Fictitious Vertical Line Passing Through the Heel.
Set the Weight of Soil to be the Vertical Component of the Lateral Pressure
Resolving the Lateral Pressure into Horizontal and Vertical Components
Computation of the Hydro Static Pressure at Interfaces and Points of Zero Lateral Pressure
Computation of Total Horizontal Component at Interfaces and Points of Zero Lateral Pressure
(i.e. Horizontal Component + Hydro Static Pressure)


Computation of Horizontal Component Force for Each Layer (Segment)
Computation of Horizontal Component Force for Each Layer (Segment)
Computation of Sum of All the Horizontal Component Forces
Computation of Vertical Component Force for Each Layer (Segment)
Computation of Sum of All the Vertical Component Forces
Computation of lever arm of Each Horizontal Component Force Measured from the Right Top-Most Point of the Wall
Computation of lever arm of Each Vertical Component Force Measured from the Right Top-Most Point of the Wall


Computation of Depth of Centroid ($\bar{Z}$) from the Right Top-most Point of the Wall
Computation of Shift of Centroid ($\bar{X}$) from the Right Top-most Point of the Wall
4. Output Stage
Generating Output
Graphical Output
Displaying the Horizontal and Vertical Component of the Lateral Earth Pressure
Numeric Output
Displaying the Total Horizontal and Vertical Forces of the Lateral Earth Pressure with Their Respective Depth and Shift of Centroid.
Data Entered and Computed Values at the Previous Sequence of Steps are Presented in Tabular Form.

II. Retaining Wall

1. Data Entry Stage

Selection of Type of Retaining Wall to be Analyzed/Designed (i.e. Gravity/Cantilever)

↓

Editing Default Dimensions

↓

Inputting Retaining Wall Parameters; Unit Weight of Wall, Front Face Soil Type, Thickness and Imposed Surcharge, Allowable Bearing Capacity of Base Soil, Base Soil-Structure Friction Angle, Base Soil Adhesion, Factor of Safety against Sliding and Overturning

↓

2. Analysis Stage

Computation of Weight of Wall (W_{wall})

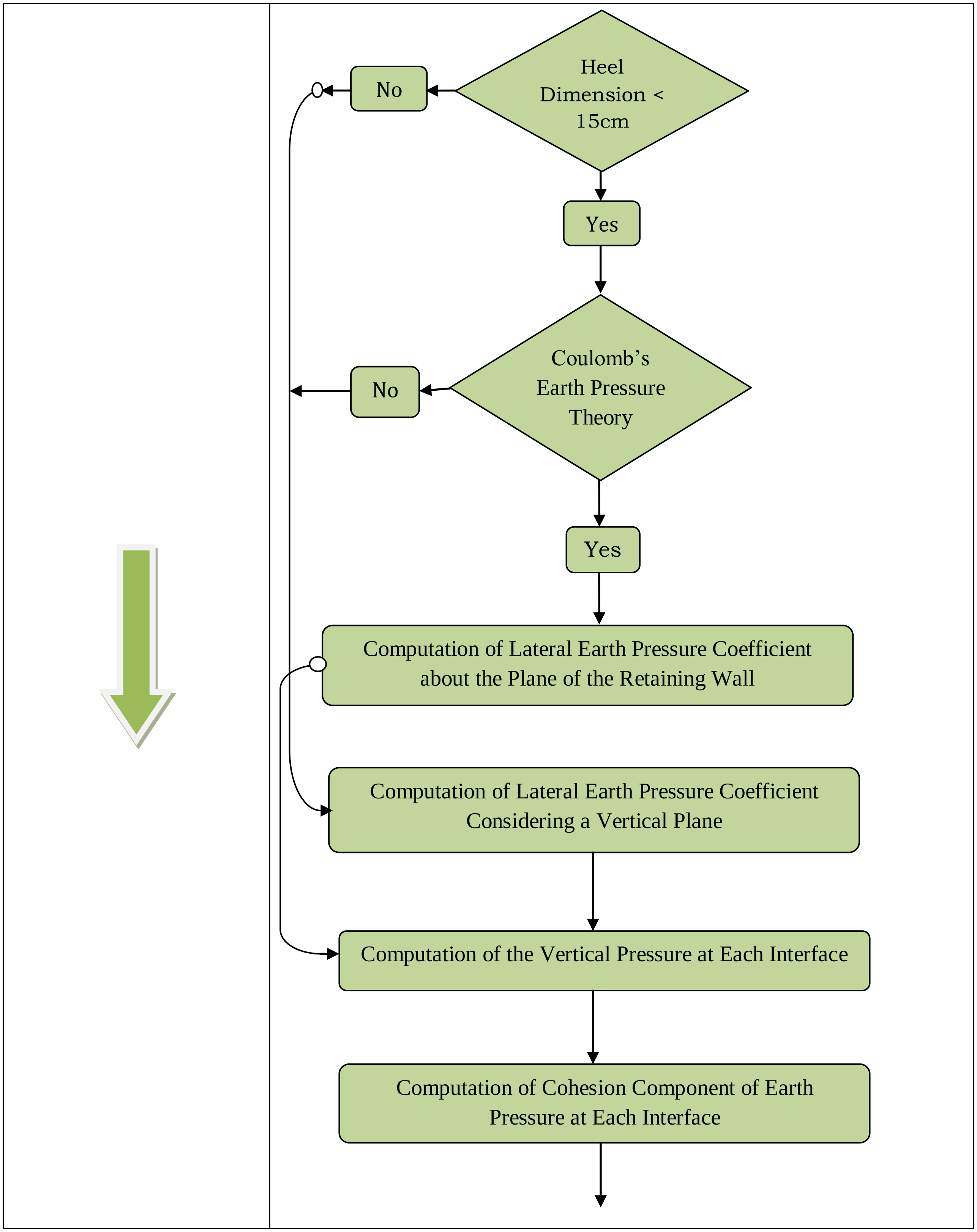
↓

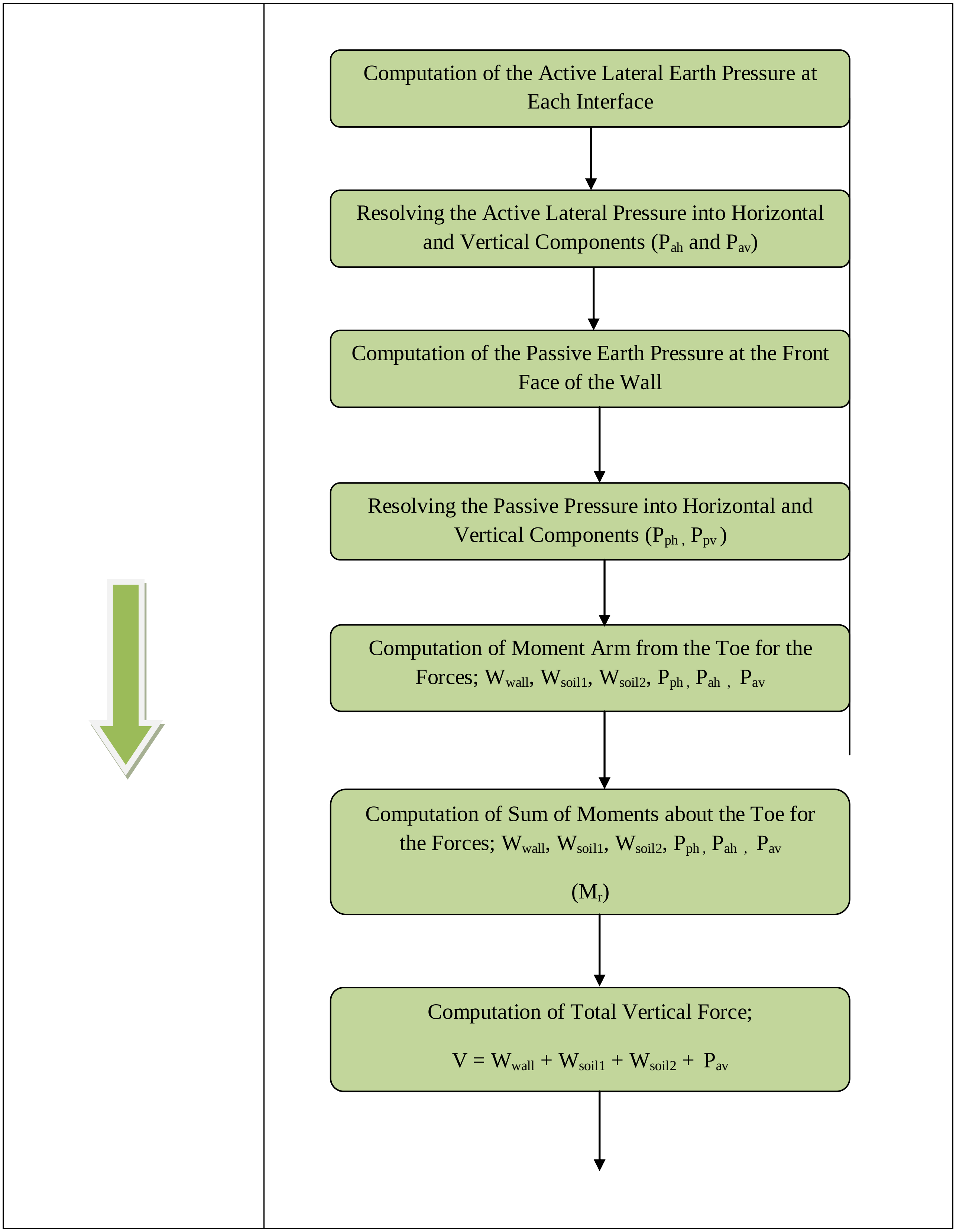
Computation of Weight of Soil Contained Between a Fictitious Vertical Plane Passing Through the Heel and the Back Wall Face Together with the Surcharge (W_{Soil1})

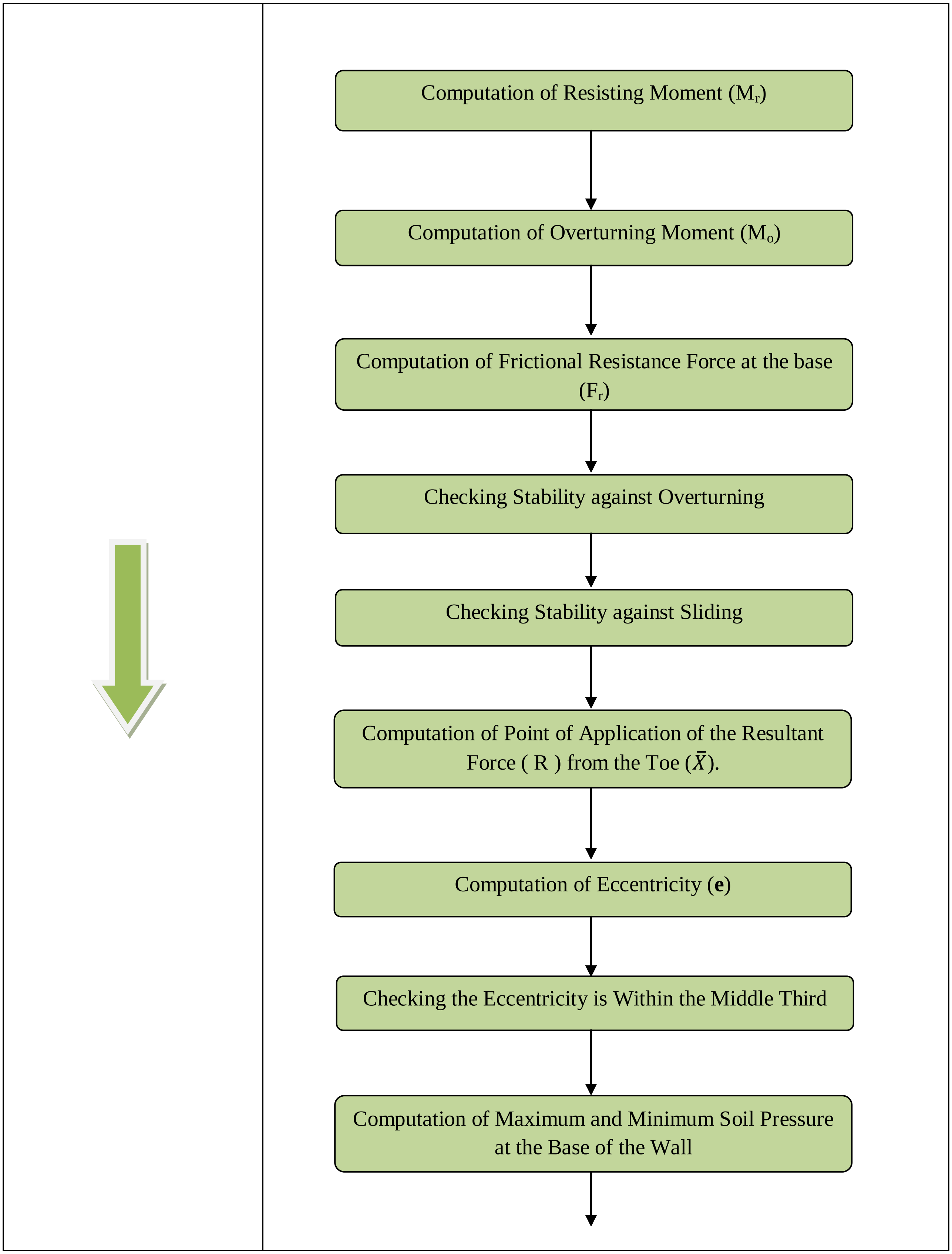
↓

Computation of Weight of Soil Resting on the Toe Together with the Surcharge (W_{Soil2})

↓


Heel Dimension < 15cm
No
Yes
Coulomb's Earth Pressure Theory
No
Yes
Computation of Lateral Earth Pressure Coefficient about the Plane of the Retaining Wall
Computation of Lateral Earth Pressure Coefficient Considering a Vertical Plane
Computation of the Vertical Pressure at Each Interface
Computation of Cohesion Component of Earth Pressure at Each Interface


Computation of the Active Lateral Earth Pressure at Each Interface
Resolving the Active Lateral Pressure into Horizontal and Vertical Components (P_{ah} and P_{av})
Computation of the Passive Earth Pressure at the Front Face of the Wall
Resolving the Passive Pressure into Horizontal and Vertical Components (P_{ph}, P_{pv})
Computation of Moment Arm from the Toe for the Forces; W_{wall}, W_{soil1}, W_{soil2}, P_{ph}, P_{ah}, P_{av}
Computation of Sum of Moments about the Toe for the Forces; W_{wall}, W_{soil1}, W_{soil2}, P_{ph}, P_{ah}, P_{av} (M_r)
Computation of Total Vertical Force;
$V = W_{wall} + W_{soil1} + W_{soil2} + P_{av}$

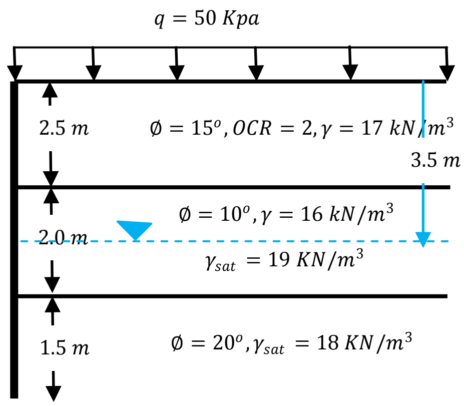

Computation of Resisting Moment (M_r)
Computation of Overturning Moment (M_o)
Computation of Frictional Resistance Force at the base (F_r)
Checking Stability against Overturning
Checking Stability against Sliding
Computation of Point of Application of the Resultant Force (R) from the Toe ($\bar{X}$).
Computation of Eccentricity (**e**)
Checking the Eccentricity is Within the Middle Third
Computation of Maximum and Minimum Soil Pressure at the Base of the Wall

	Checking the Maximum Soil Pressure Does not Exceed the Allowable Bearing Capacity
3. Output Stage	**Generating Output** **Graphical Output:** Showing the Dimensions of the Wall, Forces Acting at the Wall System with their Relative Location. **Numeric Output:** Forces and their Point of Application, Checking for Sliding, Overturning and Bearing Capacity Failure are Presented in Tabular Forms

ANNEX-C: - Illustrative Examples

Illustrative examples

1. Plot the at-rest earth pressure diagram and compute the resultant force and its location $\bar{z}$ from the top of the wall for the wall system shown in Fig. E 1.1.5.



Figure; E1.1.5

I. Results from hand/spread sheet calculations

- Computation of at-rest pressure coefficient for each layer:

Table; E1.1.5.: Computation of at-rest pressure Coefficient

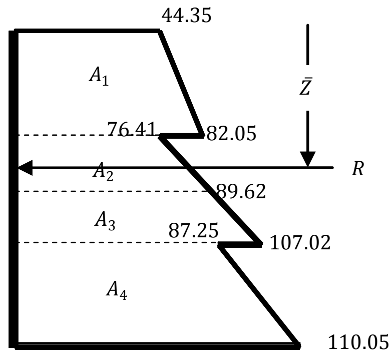
Layer No. (Top to Bottom)	Pressure coefficient	Equation No.
1	0.887	2.5
2	0.826	2.2
3	0.658	2.2

- Computation of at-rest Lateral Earth pressure:

Table; E1.2.5: Computation of at-rest Lateral Earth Pressure

Depth (m), From top to Bottom	Coefficient of Earth pressure	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Lateral Earth pressure	Hydrostatic Pressure	Total Lateral Pressure
0	0.887	50	17			9.81		44.350	0.000	44.350
2.5	0.887	50	17			9.81		82.048	0.000	82.048
2.5	0.826	50	17			9.81		76.405	0.000	76.405
3.5	0.826	50	16			9.81		89.621	0.000	89.621
4.5	0.826	50		19	1	9.81	9.19	97.212	9.810	107.022
4.5	0.658	50		19	1	9.81	9.19	77.440	9.810	87.250
6	0.658	50		18	2.5	9.81	8.19	85.524	24.525	110.049

- Lateral Earth pressure diagram:



- Computation of the Resultant Force:

$$R = \sum_{i=1}^{4} \mathrm{A_i}$$
$$= 0.5 * (44.35 + 82.05) * 2.5 + 0.5 * (76.41 + 89.62) * 1 + 0.5 * (89.62 + 107.02) * 1 + 0.5 * (87.25 + 110.05) * 1.5$$

$$= \mathbf{487.31\ KN/m}$$

- Depth of Resultant Force:

$$\bar{Z} = \frac{\sum_{i=1}^{4} \mathrm{A_i\,Z_i}}{\sum_{i=1}^{4} \mathrm{A_i}}$$

$$= \frac{\mathbf{1643.155}}{\mathbf{487.31}} = \mathbf{3.372m}$$

II. Results from the developed software

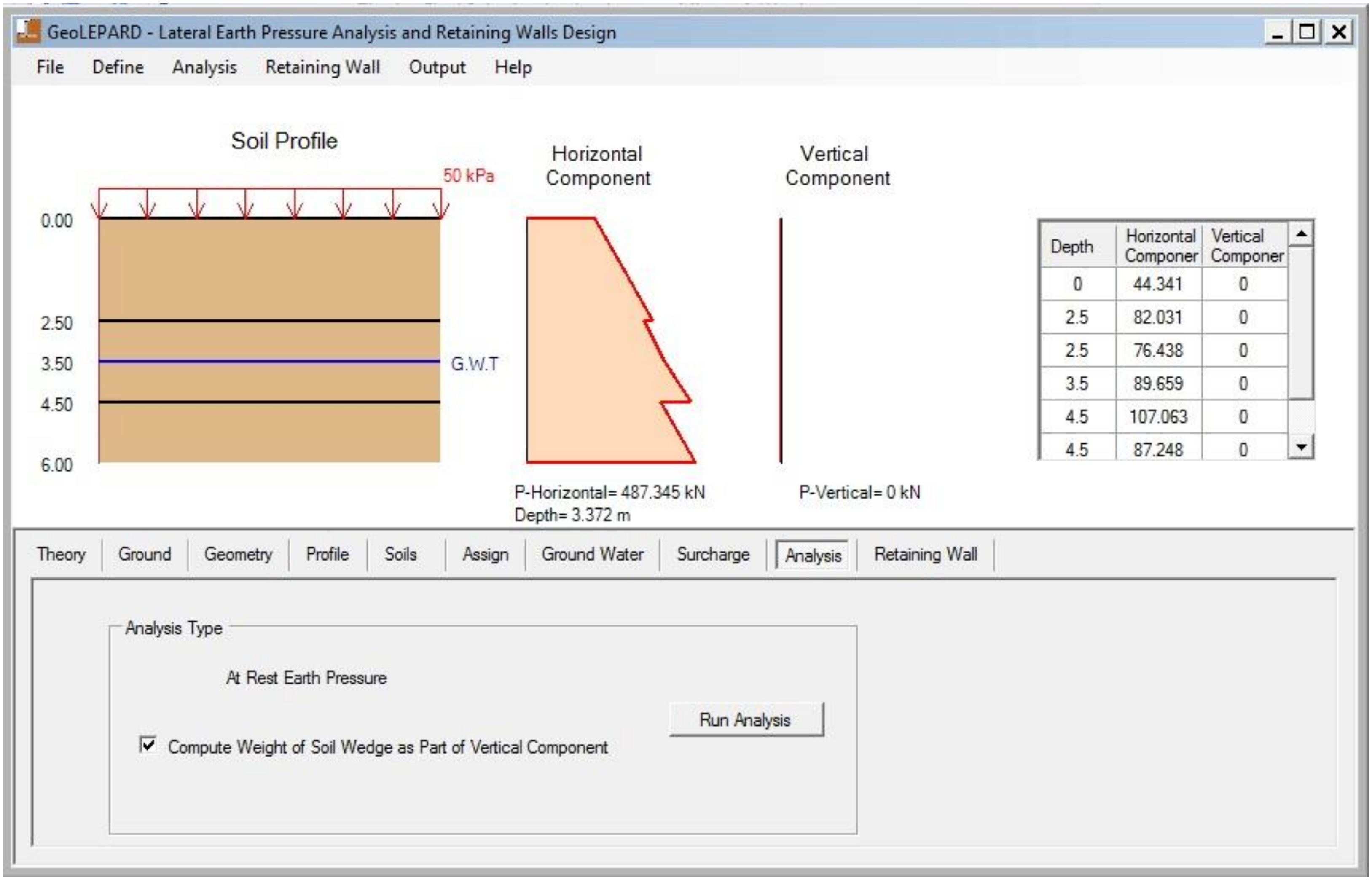


Figure: E1.2.5 Graphical & tabular Solution from the developed software

2. Plot the at-rest earth pressure diagram and compute the resultant force, R and its location $\bar{z}$ from the top of the wall for the wall system shown in Fig. E 2.1.5.

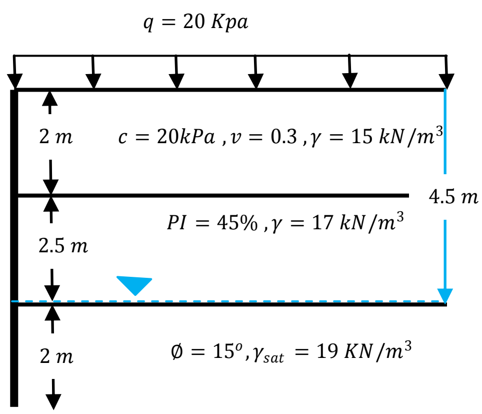


Figure: E2.1.5:

I. Results from hand/spread sheet calculations

➢ Computation of at-rest pressure coefficient for each layer:

Table: E2.1.5; Computation of at-rest pressure Coefficient

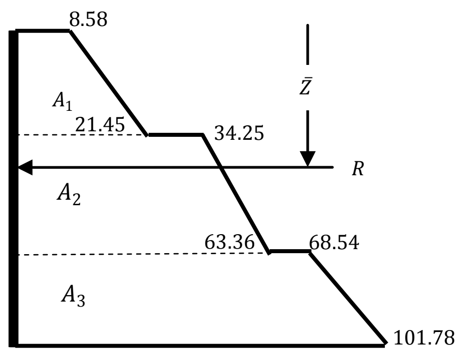
Layer No. (Top to Bottom)	Pressure coefficient	Equation No.
1	0.429	2.7
2	0.685	2.4
3	0.741	2.2

- Computation of at-rest Lateral Earth pressure:

Table: E2.2.5: Computation of at-rest Lateral Earth Pressure

Depth (m), From top to Bottom	Coefficient of Earth pressure	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT	Water unit weight	Effective Unit weight	Lateral Earth pressure	Hydrostatic Pressure	Total Lateral Pressure
0	0.429	20	15			9.81		8.580	0.000	8.580
2	0.429	20	15			9.81		21.450	0.000	21.450
2	0.685	20	15			9.81		34.250	0.000	34.250
4.5	0.685	20	17			9.81		63.363	0.000	63.363
4.5	0.741	20	17			9.81		68.543	0.000	68.543
6.5	0.741	20	19	19	2	9.81	9.19	82.162	19.620	101.782

- Lateral Earth pressure diagram:



- Computation of the Resultant Force:

$$R = \sum_{i=1}^{3} \mathrm{A_i}$$

$$= 0.5 * (8.58 + 21.45) * 2 + 0.5 * (34.25 + 63.36) * 2.5 + 0.5 * (68.54 + 101.78) * 2$$

$$= \mathbf{322.363\ kN/m}$$

- Depth of Resultant Force:

$$\bar{Z} = \frac{\sum_{i=1}^{3} \mathrm{A_i}\,\mathrm{Z_i}}{\sum_{i=1}^{3} \mathrm{A_i}}$$

$$= \frac{\mathbf{1393.86}}{\mathbf{322.363}} = \mathbf{4.324m}$$

II. Results from the Developed Software

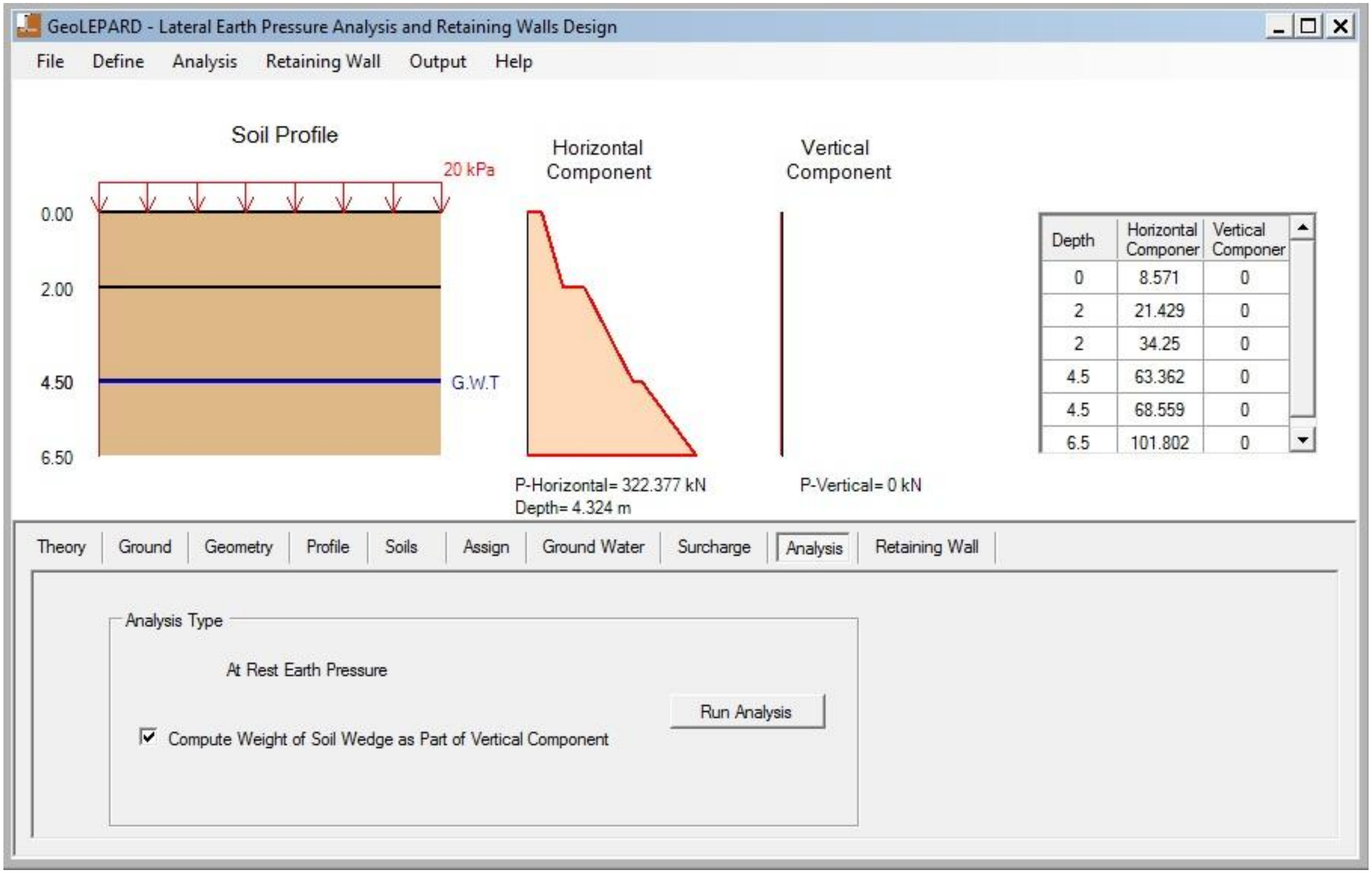


Figure: E2.2.5: Graphical and tabular Solution from the developed software

3. Plot the at-rest earth pressure diagram and compute the resultant force, R and its location $\bar{z}$ from the top of the wall for the wall system shown in Fig. E 3.1.5.

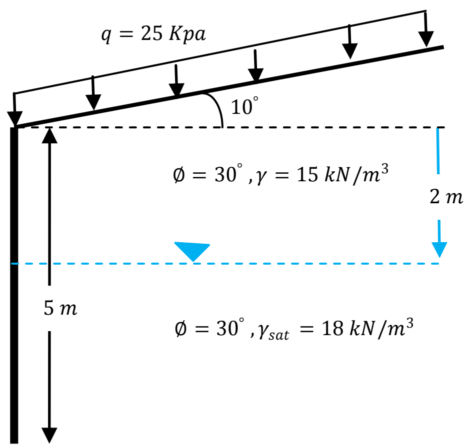


Figure: E3.1.5

I. <u>**Results from hand/spread sheet calculations**</u>

- Computation of at-rest pressure coefficient for each layer:

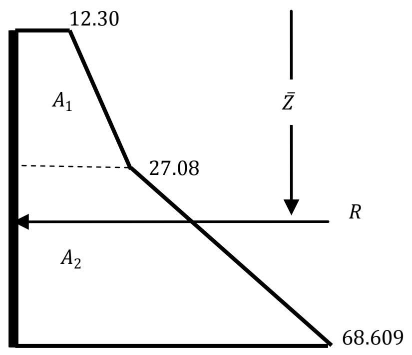
$k_{0,\beta} = 0.5$ From Eqn: 2.8

- Computation of at-rest Lateral Earth pressure:

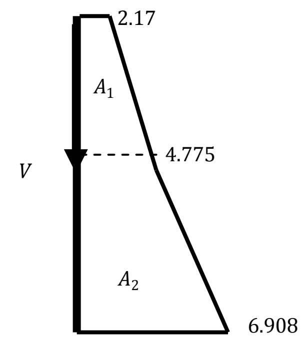
Table: E 3.1.5: Computation of at-rest Lateral Earth Pressure

Depth (m), From top to Bottom	Coefficient of Earth pressure k_0	Coefficient of Earth pressure $k_{0,\beta}$	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT	Water unit weight	Effective Unit weight	Lateral Earth pressure	Hydrostatic Pressure	Total Lateral Pressure	Horizontal Component	Vertical Component
0	0.426	0.5	25	15			9.81		12.499	0	12.499	12.309	2.170
2	0.426	0.5	25	15			9.81		27.499	0	27.499	27.081	4.775
5	0.426	0.5	25		18	3	9.81	8.19	39.783	29.43	69.213	68.609	6.908

- Lateral Earth pressure diagram:



Horizontal component



Vertical component

- Computation of the Resultant Force:

Horizontal Component

$$R = \sum_{i=1}^{3} \mathrm{A_i} \quad = 0.5 * (12.3 + 27.08) * 2 + 0.5 * (27.08 + 68.609) * 3$$

$$= \mathbf{182.924\ KN/m}$$

- Depth of Resultant Force:

$$\bar{Z} = \frac{\sum_{i=1}^{4} \mathrm{A_i\, Z_i}}{\sum_{i=1}^{4} \mathrm{A_i}}$$

$$= \frac{\mathbf{577.829}}{\mathbf{182.924}} = \mathbf{3.159\ m}$$

Vertical Component

$$V = \sum_{i=1}^{3} \mathrm{A_i} \quad = 0.5 * (2.17 + 4.775) * 2 + 0.5 * (4.775 + 6.908) * 3$$

$$= \mathbf{24.471\ KN/m}$$

- **Shift of Centroid**;

$$\bar{X} = 0$$

II. Results from the developed software

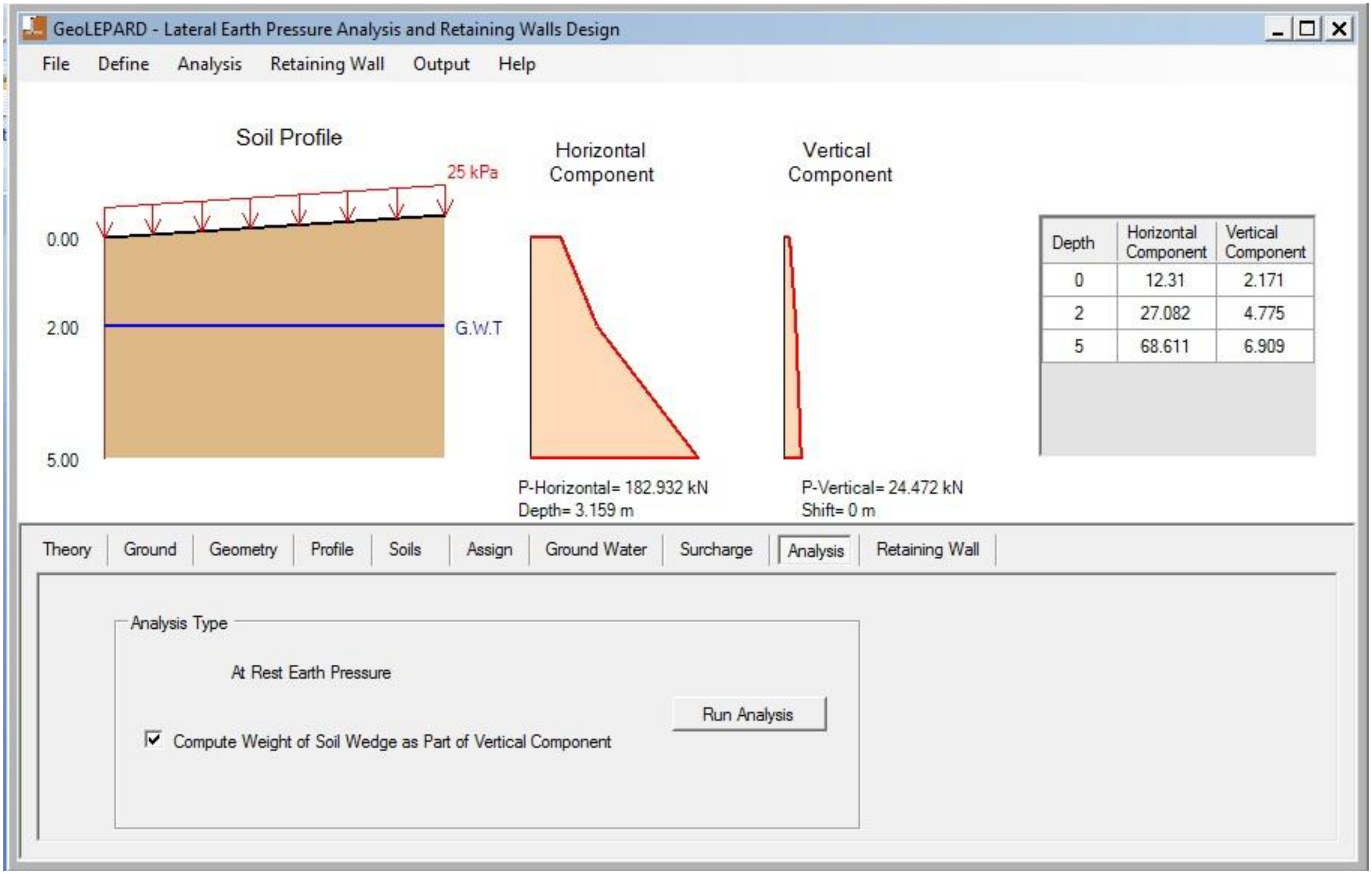


Figure: E3.2.5: Graphical and tabular Solution from the developed software

4. Re-do Example-1 for a wall having a battering of 10^0 from the vertical.

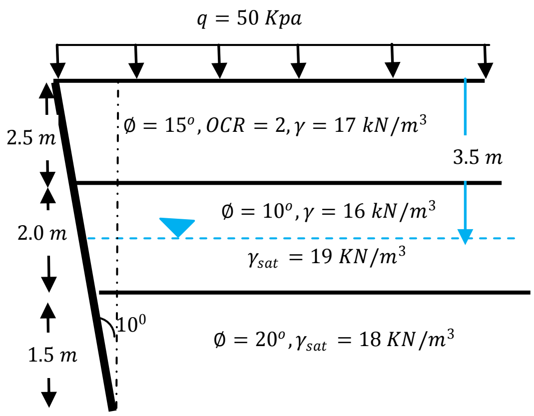


Figure: E4.1.5

I. Results from hand/spread sheet calculations

➢ Computation of at-rest pressure coefficient for each layer:

Table: E4.1.5: Computation of at-rest pressure Coefficient

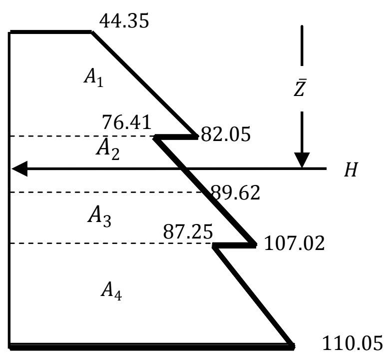
Layer No. (Top to Bottom)	Pressure coefficient	Equation No.
1	0.887	2.6
2	0.826	2.2
3	0.658	2.2

- Computation of horizontal component Lateral Earth pressure:

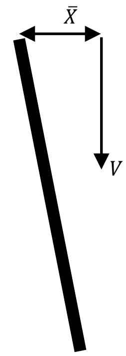
Table E4.2.5: Computation of Horizontal component of at-rest Lateral Earth Pressure

Depth (m), From top to Bottom	Coefficient of Earth pressure	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Lateral Earth pressure	Hydrostatic Pressure	Total Lateral Pressure
0	0.887	50	17			9.81		44.350	0.000	44.350
2.5	0.887	50	17			9.81		82.048	0.000	82.048
2.5	0.826	50	17			9.81		76.405	0.000	76.405
3.5	0.826	50	16			9.81		89.621	0.000	89.621
4.5	0.826	50		19	1	9.81	9.19	97.212	9.810	107.022
4.5	0.658	50		19	1	9.81	9.19	77.440	9.810	87.250
6	0.658	50		18	2.5	9.81	8.19	85.524	24.525	110.049

- Lateral Earth pressure diagram:



Horizontal Component



Vertical Component

- Computation of total Horizontal force

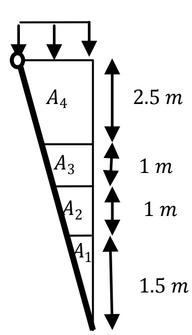
$$H = \sum_{i=1}^{4} \mathrm{A_i}$$

$= 0.5 * (44.35 + 82.05) * 2.5 + 0.5 * (76.41 + 89.62) * 1 + 0.5 * (89.62 + 107.02) * 1 + 0.5 * (87.25 + 110.05) * 1.5$

$= \mathbf{487.31\ KN/m}$

- Vertical component of Lateral Earth pressure

Vertical component of the lateral earth pressure equals the weight of the wedge of soil formed.



$$Total\ weight\ of\ wedge = V = \left(\sum_{i=1}^{4} A_i \gamma_i\right) + qH \tan 10$$

$= 0.198 * 18 + 0.353 * 19 + 0.529 * 16 + 2.099 * 17 + 50 * 6 * tan10$
$= \mathbf{107.229\ kN/m}$

- Computation of Depth of Centroid and Shift of Centroid:

  The depth of centroid of the horizontal component of the pressure is the same as that of the depth of centroid of Example-1.

- Depth of Horizontal component:

$$\bar{Z} = \frac{\sum_{i=1}^{4} \mathrm{A_i\, Z_i}}{\sum_{i=1}^{4} \mathrm{A_i}}$$

$$= \frac{\mathbf{1643.155}}{\mathbf{487.31}} = \mathbf{3.372m}$$

- The shift of centroid can be computed by taking moment about point "O".

$$\bar{\boldsymbol{X}} = \frac{\sum \boldsymbol{M_o}}{\sum \boldsymbol{F}}$$

$$= \frac{\mathbf{66.285}}{\mathbf{107.229}} = \mathbf{0.618\ m}$$

II. Results from the developed software

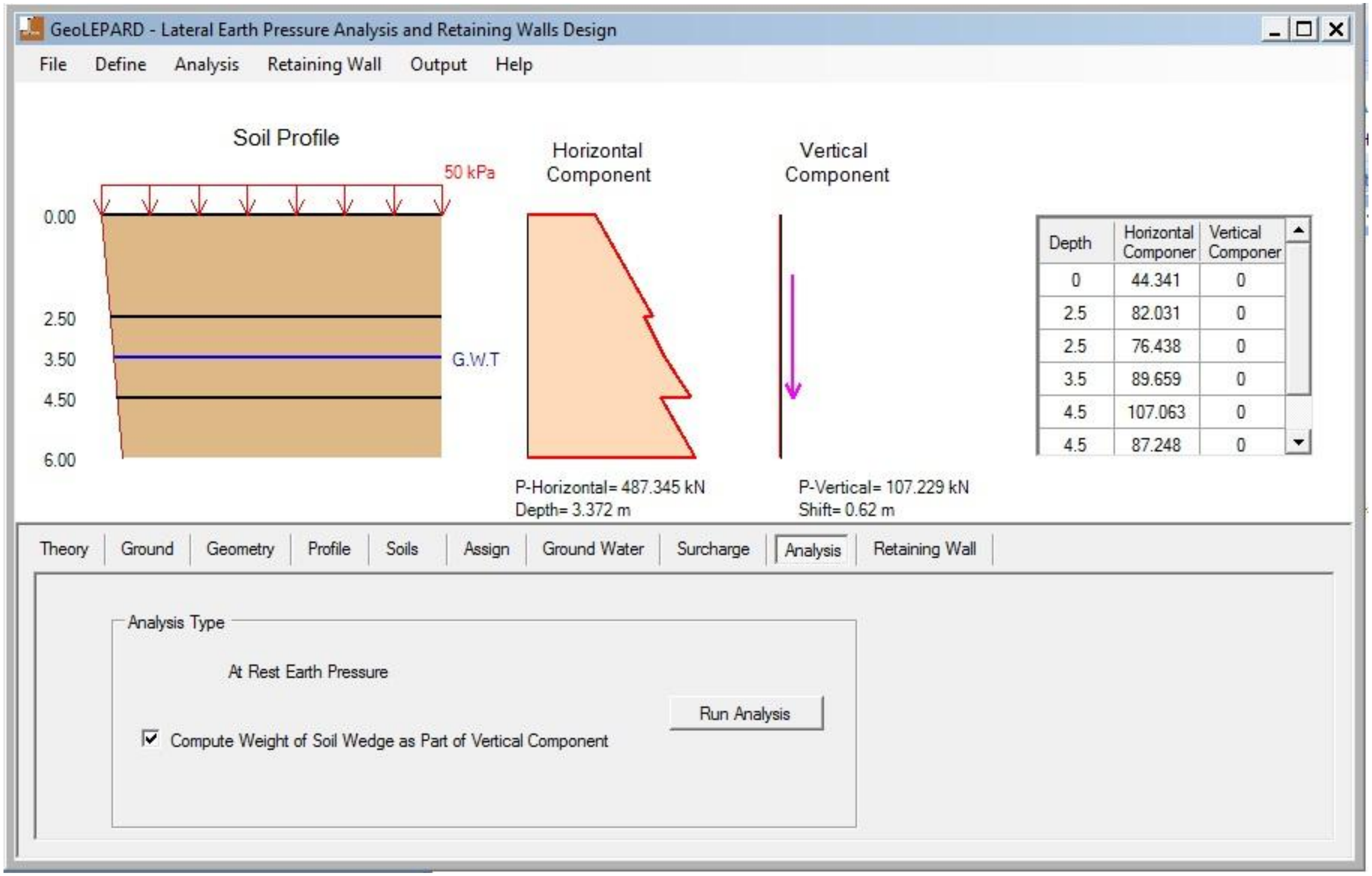


Figure: E4.2.5: Graphical and tabular Solution from the developed software

5. Re-do Example-3 for a wall having a battering of 20^0 from the vertical.

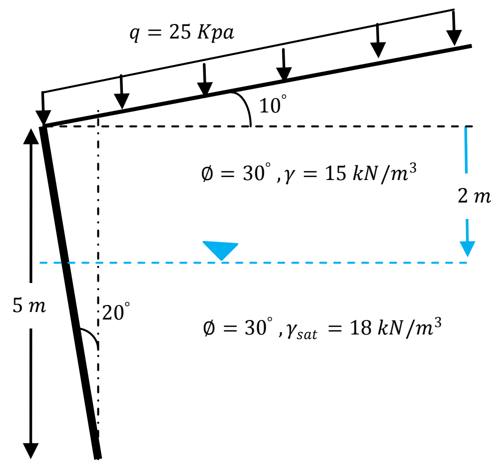


Figure: E5.1.5

I. Results from hand/spread sheet calculations

- Computation of at-rest pressure coefficient:

$k_{o,\beta} = 0.5$ From Eqn: 2.8

- Computation of Horizontal Component of Lateral Earth pressure:

  To compute the Horizontal component of the earth pressure we draw a fictitious vertical line through the heel of the wall towards the ground.

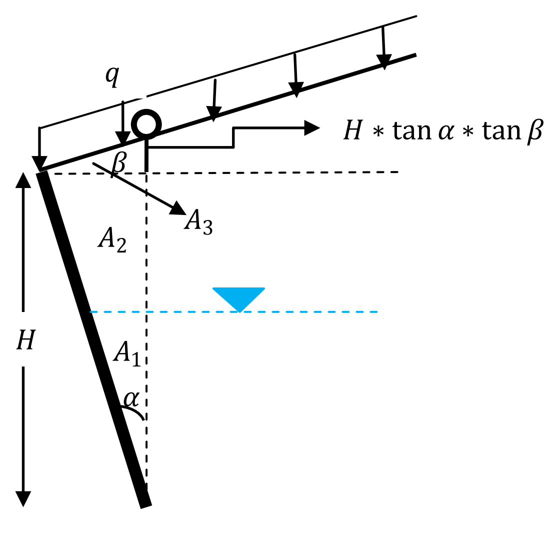


Figure: E5.2.5

$$\boldsymbol{N.B: In\ this\ case, height\ of\ consideration\ shall\ be: (H + H * tan\alpha * tan\beta)}$$

- **Vertical component of Lateral Earth pressure**

Vertical component of the lateral earth pressure equals the weight of the wedge of soil formed.

$$Total\ weight\ of\ wedge(including\ surcharge) = V_1 = \left(\sum_{i=1}^{3} A_i\gamma_i\right) + qH\left[\frac{\tan\alpha}{\cos\beta}\right]$$

$$= 1.638 * 18 + 2.912 * 15 + 0.292 * 15 + 46.2$$

$$V_1 = \mathbf{123.74\ kN/m}$$

<u>Shift of Centroid</u>:

$$\overline{X_1} = \frac{weight\ of\ soil * \bar{x}_1 + total\ surcharge * \bar{x}_2}{V}$$

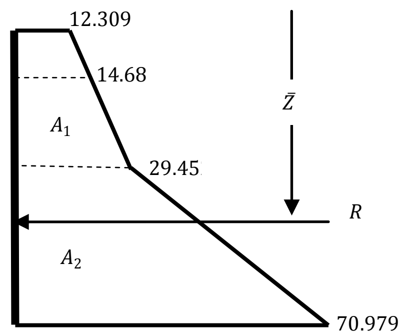
$$\overline{X_1} = \frac{77.544 * 1.213 + 46.198 * 0.91}{123.742}$$

$$\overline{X_1} = 1.1m$$

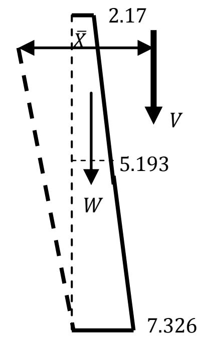
Table: E5.1.5: Computation of at-rest Lateral Earth Pressure

Depth (m), From top to Bottom	Coefficient of Earth pressure k_0	Coefficient of Earth pressure $k_{0,\beta}$	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT	Water unit weight	Effective Unit weight	Lateral Earth pressure	Hydrostatic Pressure	Total Lateral Pressure	Horizontal Component
0	0.426	0.5	25	15			9.81		12.499		12.309	2.170
0.321	0.426	0.5	25	15			9.81		14.907		14.680	2.589
2.321	0.426	0.5	25	15			9.81		29.906		29.452	5.193
5.321	0.426	0.5	25		18	3	9.81	8.19	42.190	29.430	70.979	7.326

- **Lateral Earth pressure diagram:**



Horizontal Component



- Computation of the Resultant Force:

Horizontal component

$$R = \sum_{i=1}^{3} \mathrm{A_i} \quad = 0.5 * (12.309 + 29.452) * 2.321 + 0.5 * (29.452 + 70.979) * 3.321$$

$$= \mathbf{199.112\ KN/m}$$

- Depth of Horizontal Force:

$$\bar{Z} = \frac{\sum_{i=1}^{2} \mathrm{A_i\, Z_i}}{\sum_{i=1}^{2} \mathrm{A_i}}$$

$$= \frac{\mathbf{670.3}}{\mathbf{199.11}} = \mathbf{3.367\ m}$$ **, Measured From point "o"**

$$= \mathbf{3.367 - 0.321 = 3.046m}$$**, Measured from top-most point of the wall**

Vertical component

$$V_2 = \sum_{i=1}^{3} \mathrm{A_i} \quad = 0.5 * (2.170 + 5.193) * 2.321 + 0.5 * (5.193 + 7.326) * 3.321$$

$$= \mathbf{27.325\ KN/m}$$

$$V = V_1 + V_2$$
$$V = 123.74 + 27.325$$
$$V = \mathbf{151.067 KN/m}$$

- **Shift Centroid (Pressure Diagram):**

$$\bar{X}_2 \ = H \tan\alpha$$
$$= 1.82\text{m}$$

- **Shift of Centroid;**

$$\bar{\boldsymbol{X}} = \frac{V_1\bar{X}_1 + V_2\bar{X}_2}{V_1 + V_2}$$

$$\bar{\boldsymbol{X}} = \frac{123.74 * 1.1 + 27.325 * 1.182}{123.74 + 27.325}$$

$$\bar{\boldsymbol{X}} = \mathbf{1.23}\boldsymbol{m}$$

II. Results from the developed software

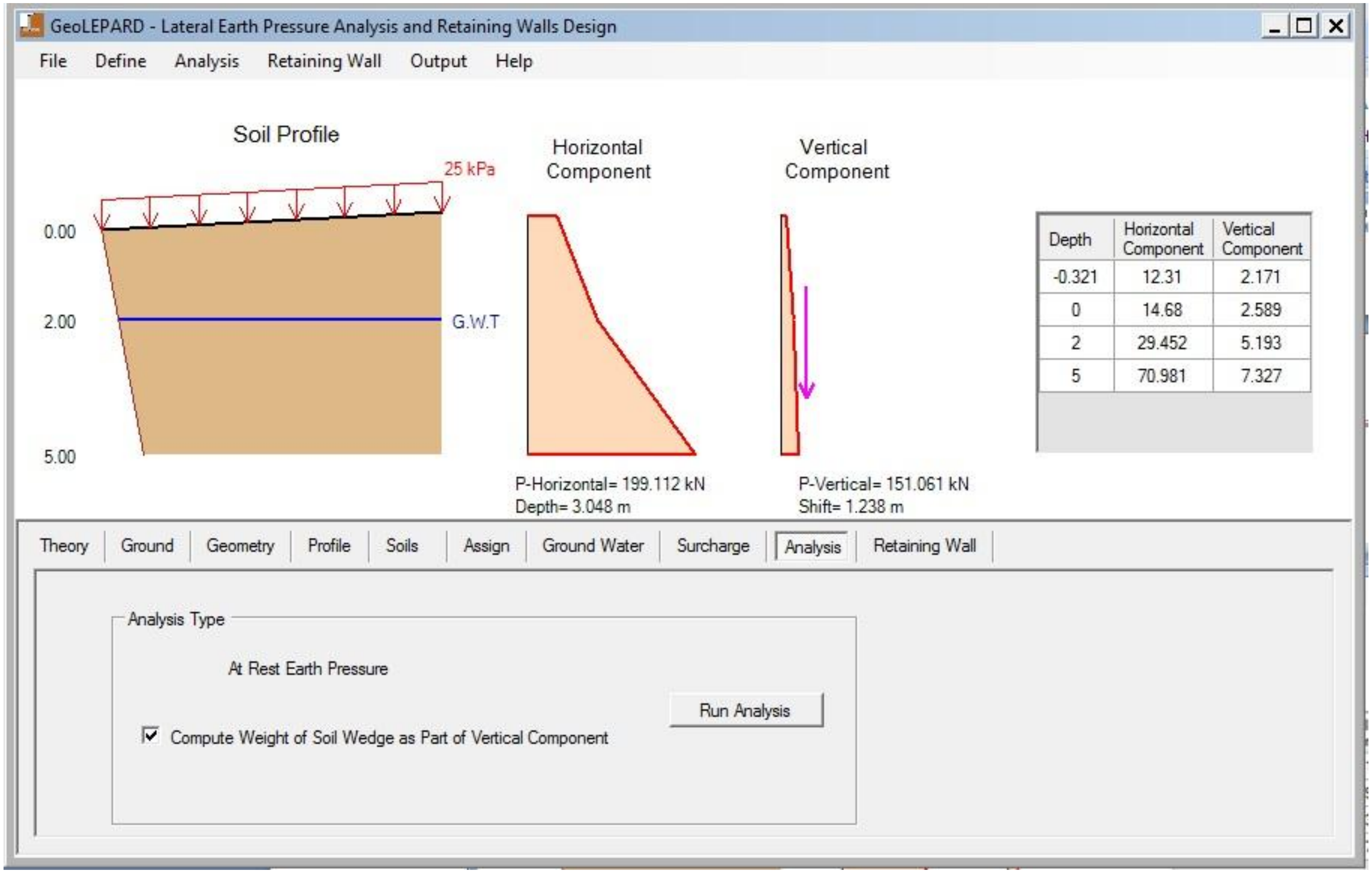


Figure: E5.3.5: Graphical and tabular Solution from the developed software

6. Plot the active and passive earth pressure diagrams and compute the horizontal and vertical components of the resultant force together with its point of application for the wall system shown in Fig. E 6.1.5, using the following Earth Pressure Theories :

   a. Rankine (Bell)
   b. Coulomb's
   c. Kerisel and Absi

Assume: $\delta = \frac{2}{3}\emptyset, and\ c_w = 0.5c$

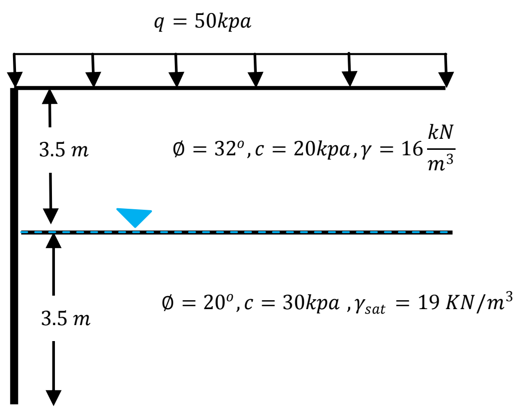


Figure: E6.1.5

a. **Rankine (Bell) Earth Pressure Theory**

I. **Results from hand/spread sheet calculations**

Coefficients of earth pressure from Eqn 2.19:

$k_{a1} = 0.307, k_{p1} = 3.255$

$k_{a2} = 0.490, k_{p2} = 2.04$

Tension crack depth: $Z_t = 1.39m$ from Eqn 2.40

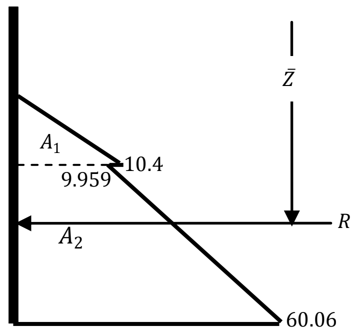
Table: E6.1.5: Computation of Active Earth Pressure

Depth (m), From top to Bottom	Coefficient of Earth pressure (K_a)	Uniform Surcharge	Cohesion	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Lateral Earth pressure	Hydrostatic Pressure	Total Lateral Pressure
0	0.3073	50	20	16			9.81		0.0	0.0	0.0
1.3851	0.3073	50	20	16			9.81		0.0	0.0	0.0
3.5	0.3073	50	20	16			9.81		10.400	0.0	10.400
3.5	0.4903	50	30	16			9.81		9.959	0.0	9.959
7	0.4903	50	30		19	3.5	9.81	9.19	26.793	34.335	60.064

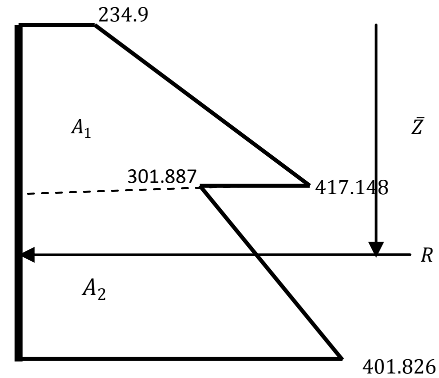
Table E6.2.5: Computation of Passive Earth Pressure

Depth (m), From top to Bottom	Coefficient of Earth pressure (K_p)	Uniform Surcharge	Cohesion	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Lateral Earth pressure	Hydrostatic Pressure	Total Lateral Pressure
0	3.255	50	20	16			9.81		234.9	0.0	234.9
3.5	3.255	50	20	16			9.81		417.148	0.0	417.148
3.5	2.04	50	30	16			9.81		301.887	0.0	301.887
7	2.04	50	30		19	3.5	9.81	9.19	367.5	34.335	401.826

- **Lateral Earth pressure diagram:**



Active Earth Pressure



Passive Earth Pressure

Total Horizontal Force(Active):

$$R = \sum_{1}^{n} \mathrm{A_i} = \mathbf{133.54\ kN/m}$$

Total Horizontal Force(Passive):

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{2372.57\ kN/m}$$

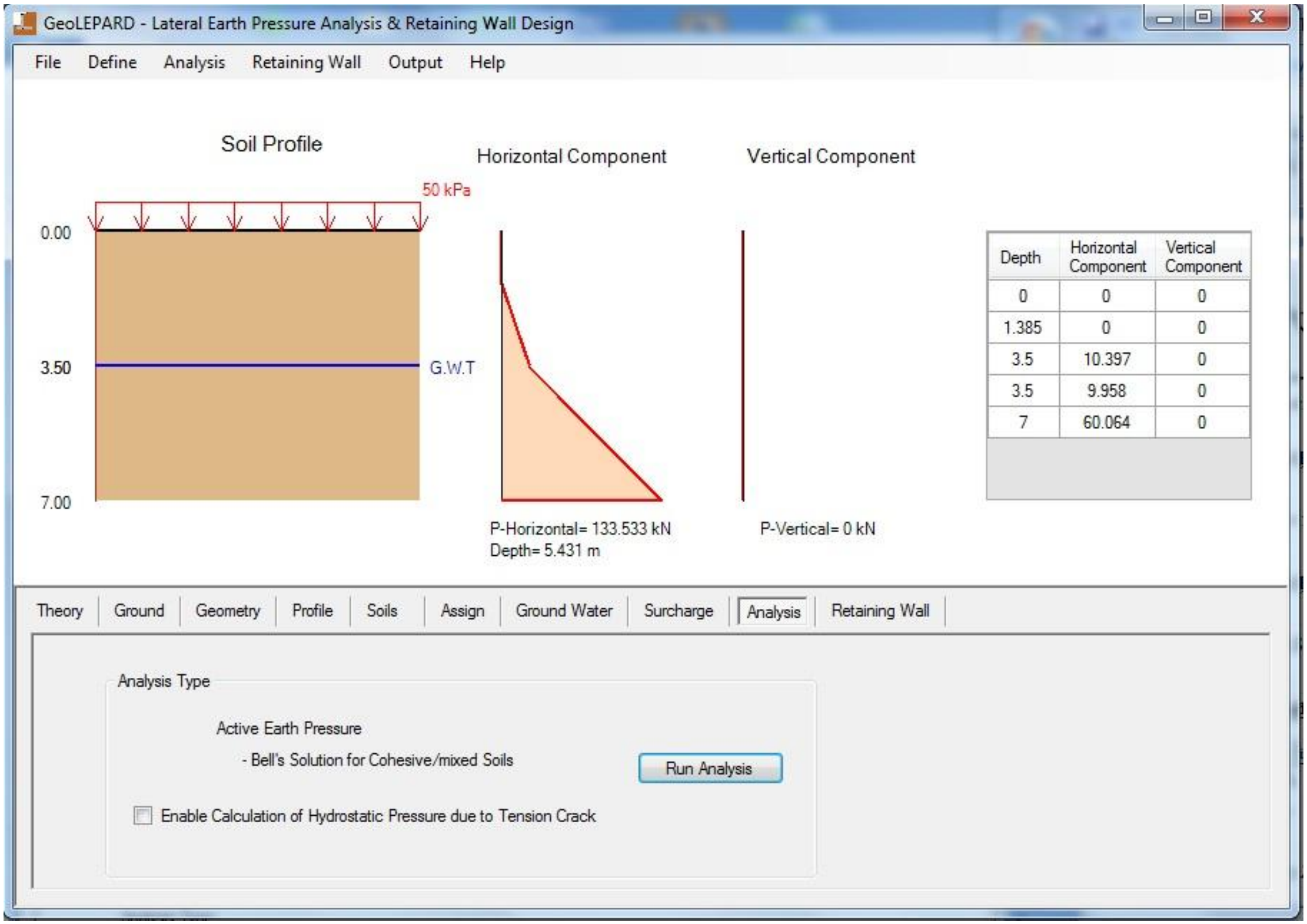
Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{2} \mathrm{A_i}\,\mathrm{Z_i}}{\sum_{i=1}^{2} \mathrm{A_i}}$$

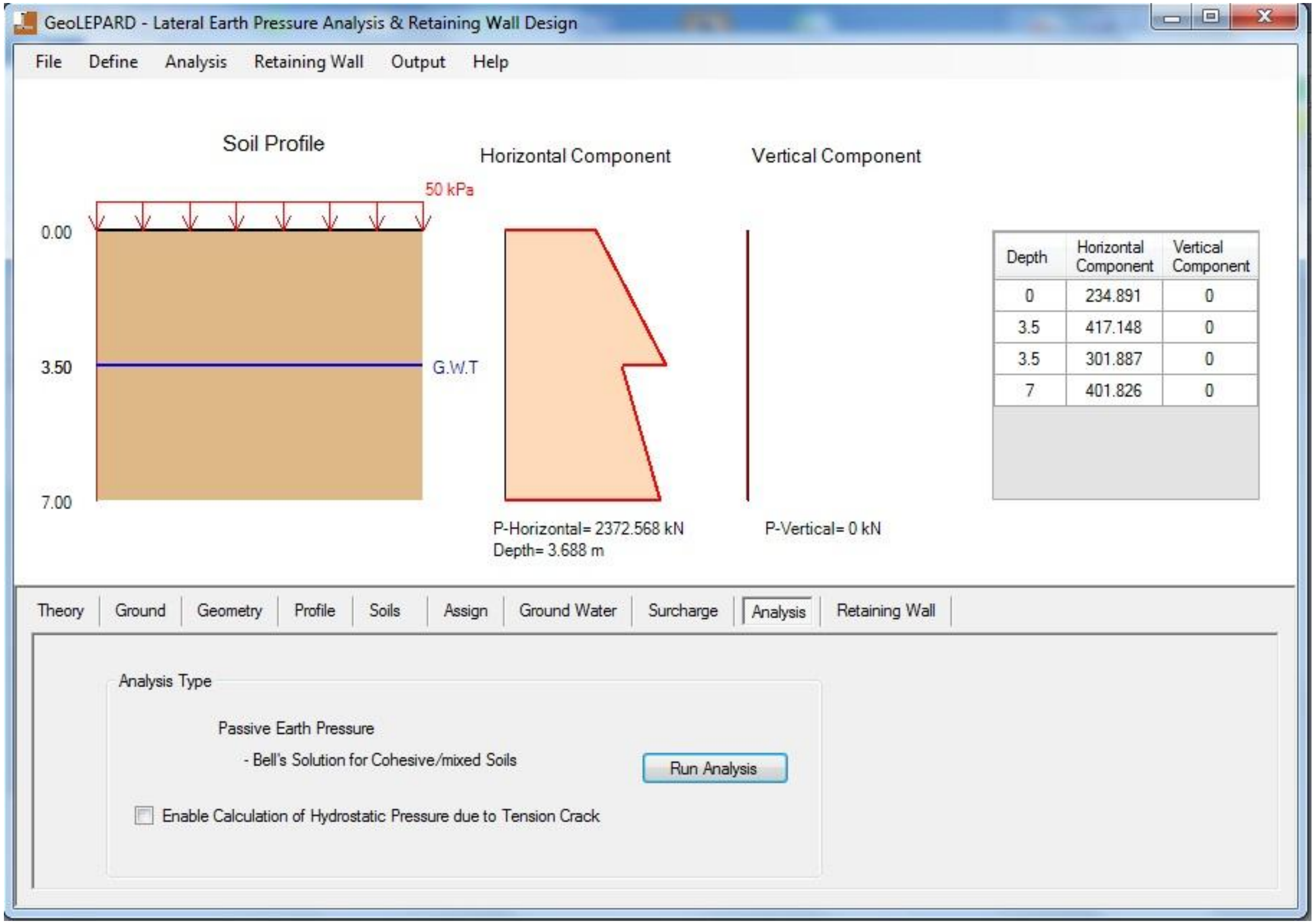
For active: $\bar{Z} = \frac{725.23}{133.54} = \mathbf{5.436\ m}$

For Pasive**:** $\bar{Z} = \frac{8750.31}{2372.57} = \mathbf{3.688\ m}$

II. Results from the developed software



(a) Active Earth Pressure



(b) Passive Earth Pressure

Figure: E6.2.5: Graphical and tabular Solution from the developed software

b. Coulomb's Earth Pressure Theory

I. Results from Hand/Spread Sheet Calculations

Coefficients of earth pressure from Eqns. 2.44 and 2.57:

$k_{a1} = 0.275, k_{p1} = 7.333, k_{ac\,1} = 1.285, k_{pc\,1} = 6.633$

$k_{a2} = 0.438, k_{p2} = 2.888, k_{ac\,2} = 1.621, k_{pc\,2} = 4.163$

Tension crack depth: $Z_t = 2.714\ m$ from Eqn. 2.55:

Table E6.3.5: Computation of Active Earth Pressure

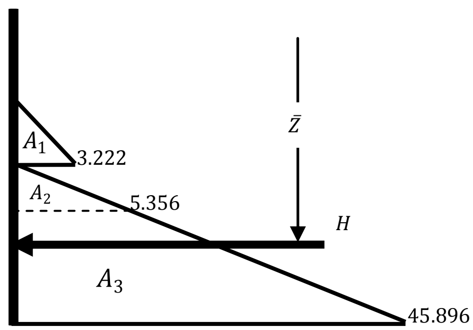
Depth (m), From top to Bottom	Coefficient of Earth pressure (Ka)	Coefficient of Earth pressure (Kac)	Uniform Surcharge	Cohesion	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	0.275	1.285	50	20	16			9.81		-11.950	0.000	0.000	0.000
2.714	0.275	1.285	50	20	16			9.81		0.000	0.000	0.000	0.000
3.5	0.275	1.285	50	20	16			9.81		3.460	0.000	3.222	1.259
3.5	0.438	1.621	50	30	16			9.81		-2.205	0.000	0.000	0.000
4.046	0.438	1.621	50	30		19	0.546	9.81	9.19	0.000	5.375	5.375	0.000
7	0.438	1.621	50	30		19	3.5	9.81	9.19	11.883	34.335	45.898	2.740

Table E6.4.5 : Computation of Passive Earth Pressure

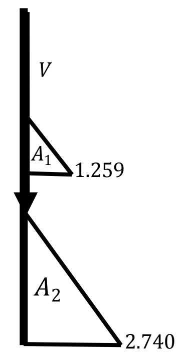
Depth (m), From top to Bottom	Coefficient of Earth pressure (Kp)	Coefficient of Earth pressure (Kpc)	Uniform Surcharge	Cohesion	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	7.333	6.633	50	20	16			9.81		499.310	0.0	465.1	181.6
3.5	7.333	6.633	50	20	16			9.81		909.958	0.0	847.6	331.0
3.5	2.888	4.163	50	30	16			9.81		431.018	0.0	419.4	99.4
7	2.888	4.163	50	30		19	3.5	9.81	9.19	523.911	34.3	544.1	120.8

- **Lateral Earth pressure diagram:**

Active Case



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{78.428\ kN/m}$$

Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{n} \mathrm{A_i\,Z_i}}{\sum_{i=1}^{n} \mathrm{A_i}}$$

$$\bar{Z} = \frac{457.316}{78.428} = \mathbf{5.831\ m}$$

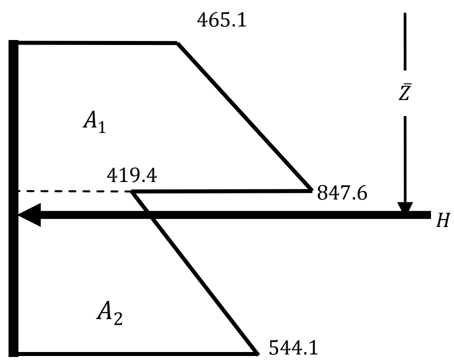
Total Vertical Force:

$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{4.539\ kN/m}$$

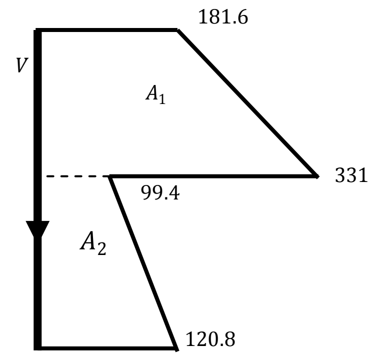
Shift of Centroid:

$$\bar{X} = 0$$

Passive Case



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{3983.35\ kN/m}$$

Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{2} \mathrm{A_i\,Z_i}}{\sum_{i=1}^{2} \mathrm{A_i}}$$

Horizontal Component: $\bar{Z} = \frac{13390.07}{3983.35} = \mathbf{3.362\ m}$

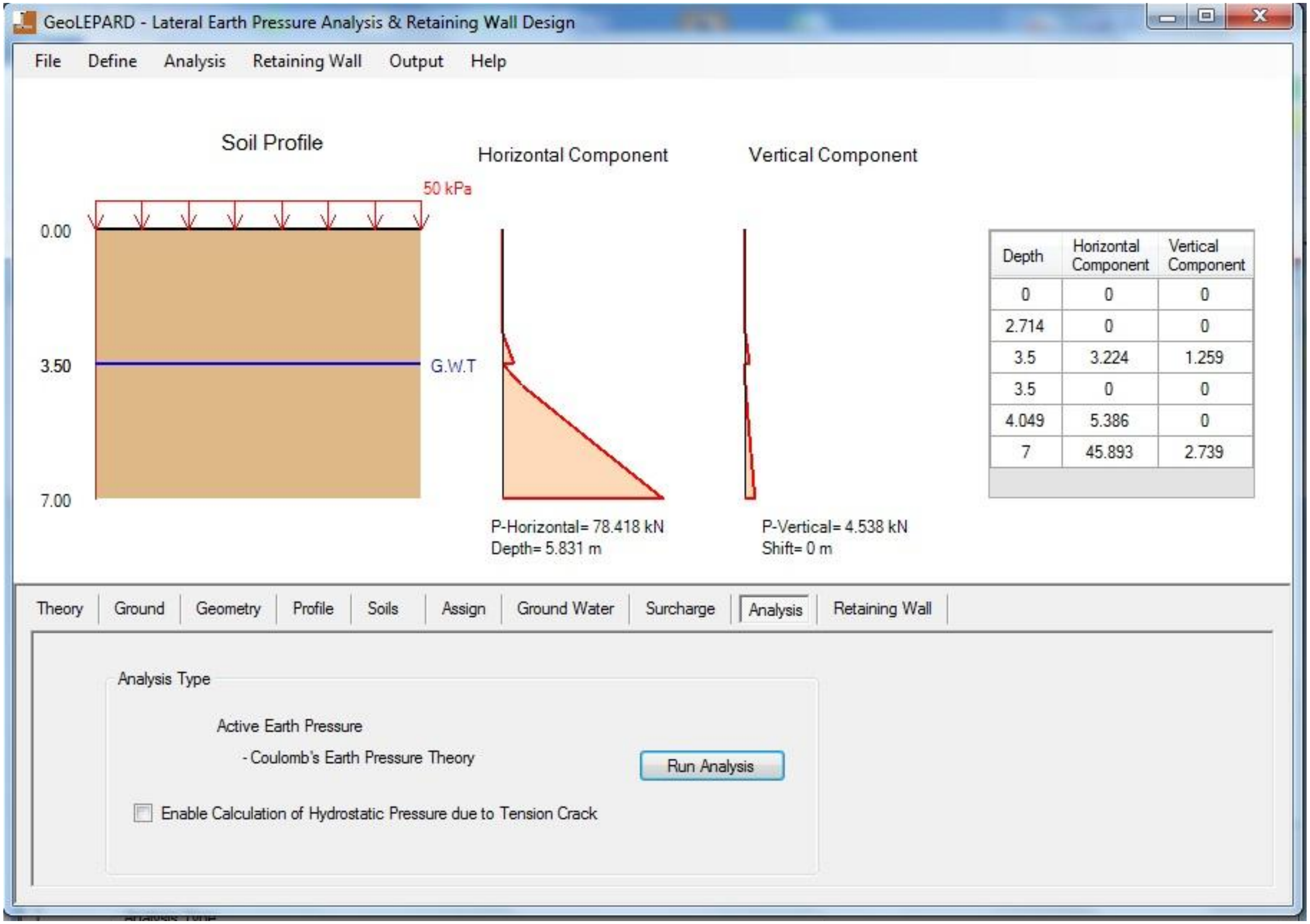
Total Vertical Force:

$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{1282.4\ kN/m}$$

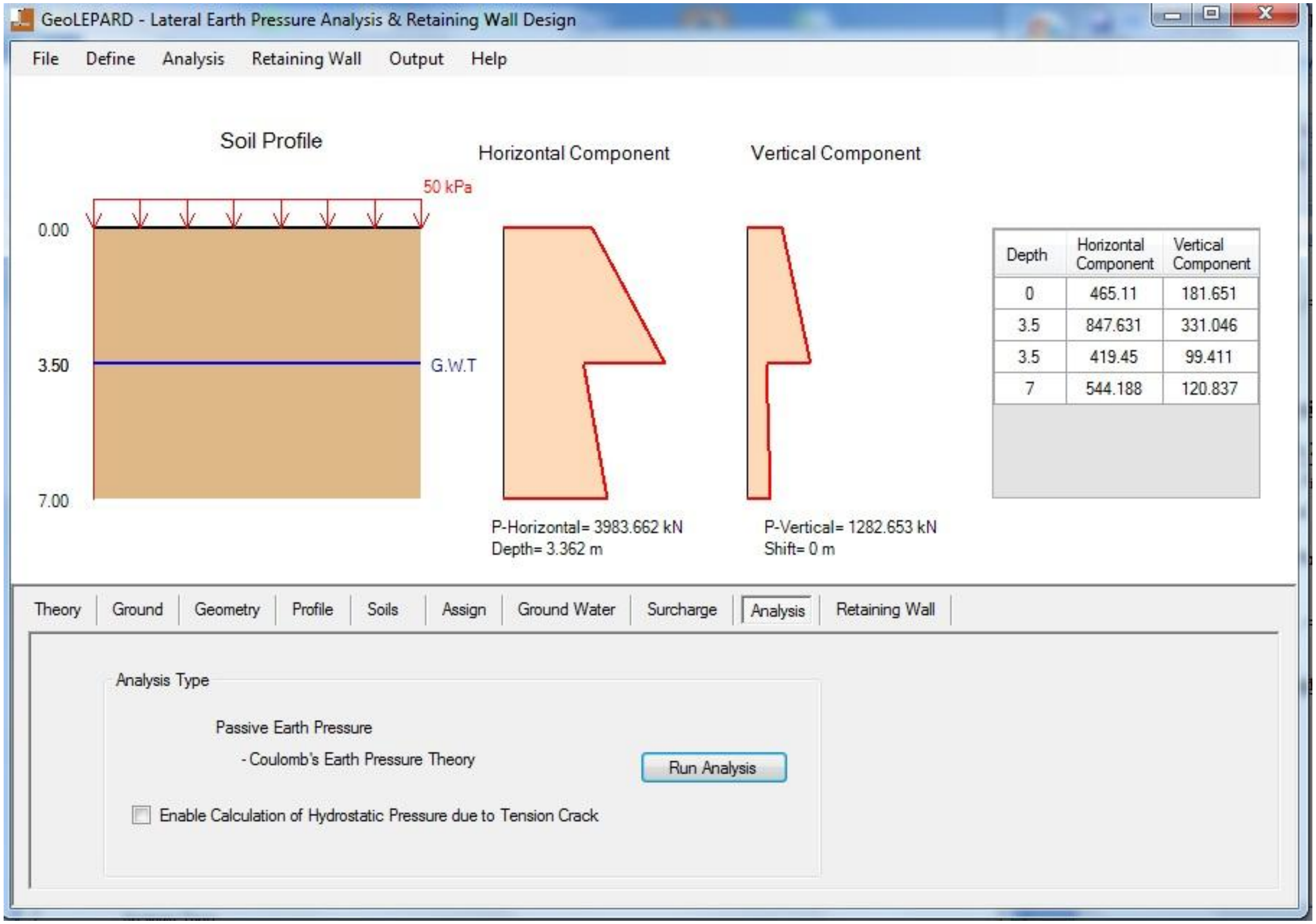
Shift of Centroid:

$$\bar{X} = 0$$

II. Results from the developed software



(a) Active earth Pressure



(b) Passive Earth Pressure

Figure: E6.3.5: Graphical and tabular Solution from the developed software

c. Kerisel and Absi Earth Pressure Theory

I. Results from Hand/Spread Sheet Calculations

Coefficients of earth pressure from Figure 2.22:

$k_{a1} = 0.252$, $k_{p1} = 6$

$k_{a2} = 0.43$, $k_{p2} = 2.7$

Tension crack depth: $Z_0 = 2.974\ m$

Table E6.5.5: Computation of Active Earth Pressure

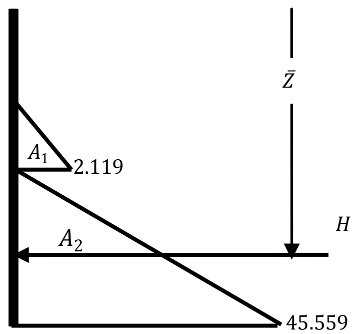
Depth (m), From top to Bottom	Coefficient of Earth pressure (Ka)	Coefficient of Earth pressure (Kac)	Uniform Surcharge	Cohesion	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	0.252	1.230	50	20	16			9.81		-11.993	0.000	0.000	0.000
2.974	0.252	1.230	50	20	16			9.81		0.000	0.000	0.000	0.000
3.5	0.252	1.230	50	20	16			9.81		2.119	0.000	2.119	0.828
3.5	0.43	1.606	50	30	16			9.81		-2.607	0.000	0.000	0.000
4.160	0.43	1.606	50	30		19	0.66	9.81	9.19	0.000	6.472	6.472	0.000
7	0.43	1.606	50	30		19	3.5	9.81	9.19	11.224	34.335	45.559	2.660

Table E6.6.5: Computation of Passive Earth Pressure

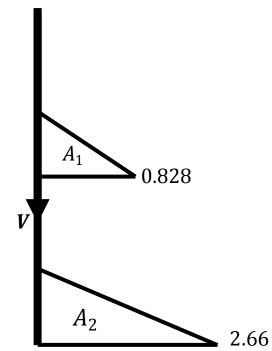
Depth (m), From top to Bottom	Coefficient of Earth pressure (Kp)	Coefficient of Earth pressure (Kpc)	Uniform Surcharge	Cohesion	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	6	6	50	20	16			9.81		420.000	0.0	420.000	164.03
3.5	6	6	50	20	16			9.81		756.000	0.0	756.000	295.26
3.5	2.7	4.025	50	30	16			9.81		406.948	0.0	406.948	96.45
7	2.7	4.025	50	30		19	3.5	9.81	9.19	493.793	34.3	528.128	117.03

- **Lateral Earth pressure diagram:**

Active Case



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{76.582\ kN/m}$$

Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{n} \mathrm{A_i\, Z_i}}{\sum_{i=1}^{n} \mathrm{A_i}}$$

Horizontal Component: $\bar{Z} = \frac{448.839}{76.582} = \mathbf{5.861\ m}$

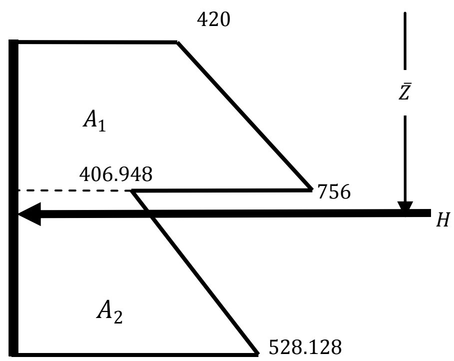
Total Vertical Force:

$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{3.995\ kN/m}$$

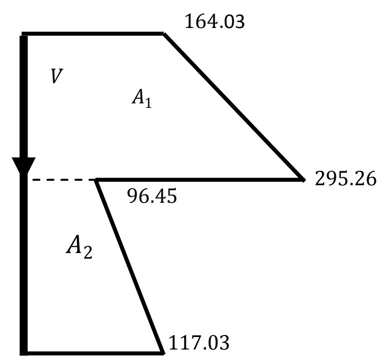
Shift of Centroid:

$$\bar{X} = 0$$

Passive Case



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{3694.38\ kN/m}$$

Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{2} \mathrm{A_i\, Z_i}}{\sum_{i=1}^{2} \mathrm{A_i}}$$

Horizontal Component: $\bar{Z} = \frac{12659.21}{3694.38} = \mathbf{3.427\ m}$

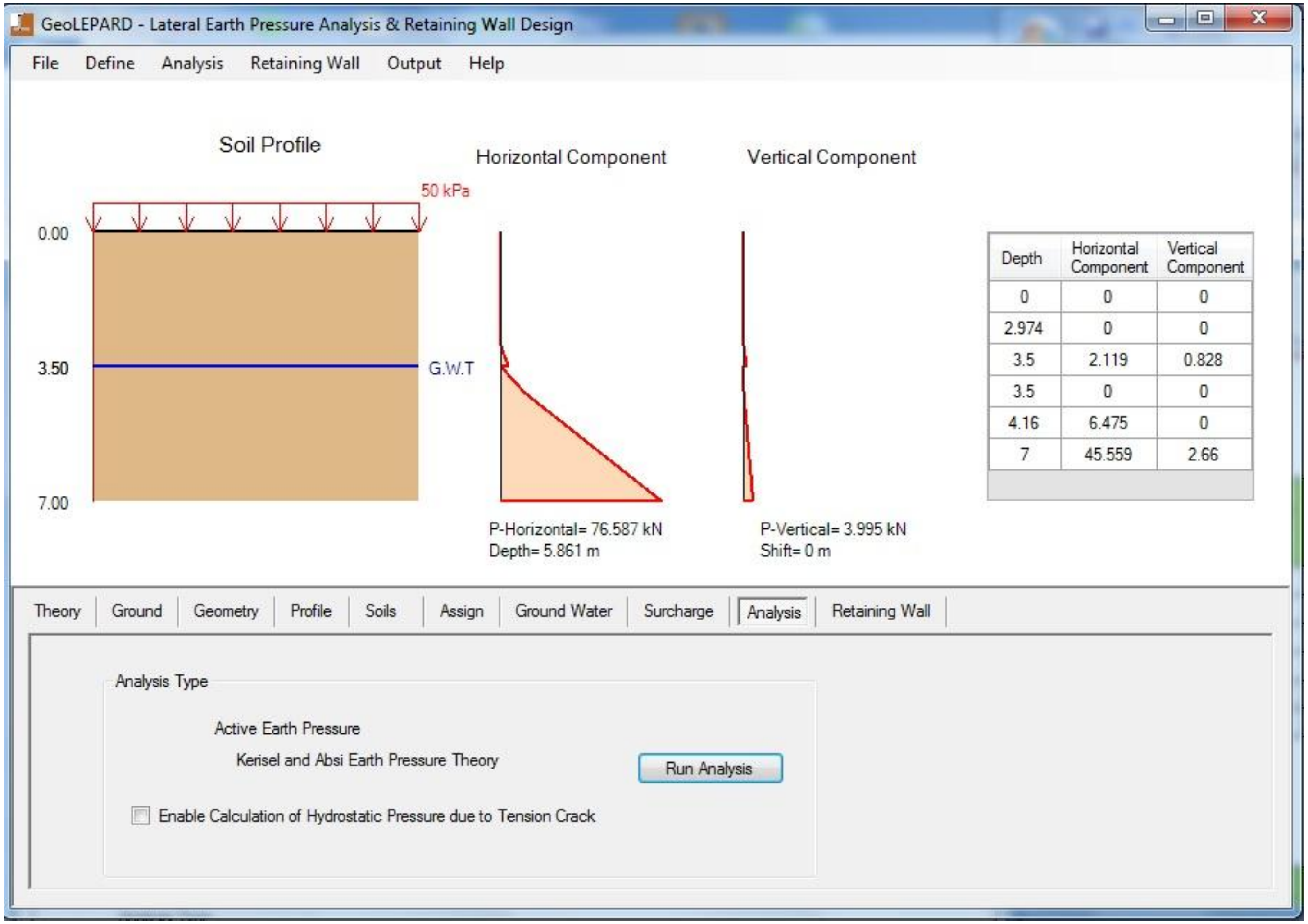
Total Vertical Force:

$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{1177.35\ kN/m}$$

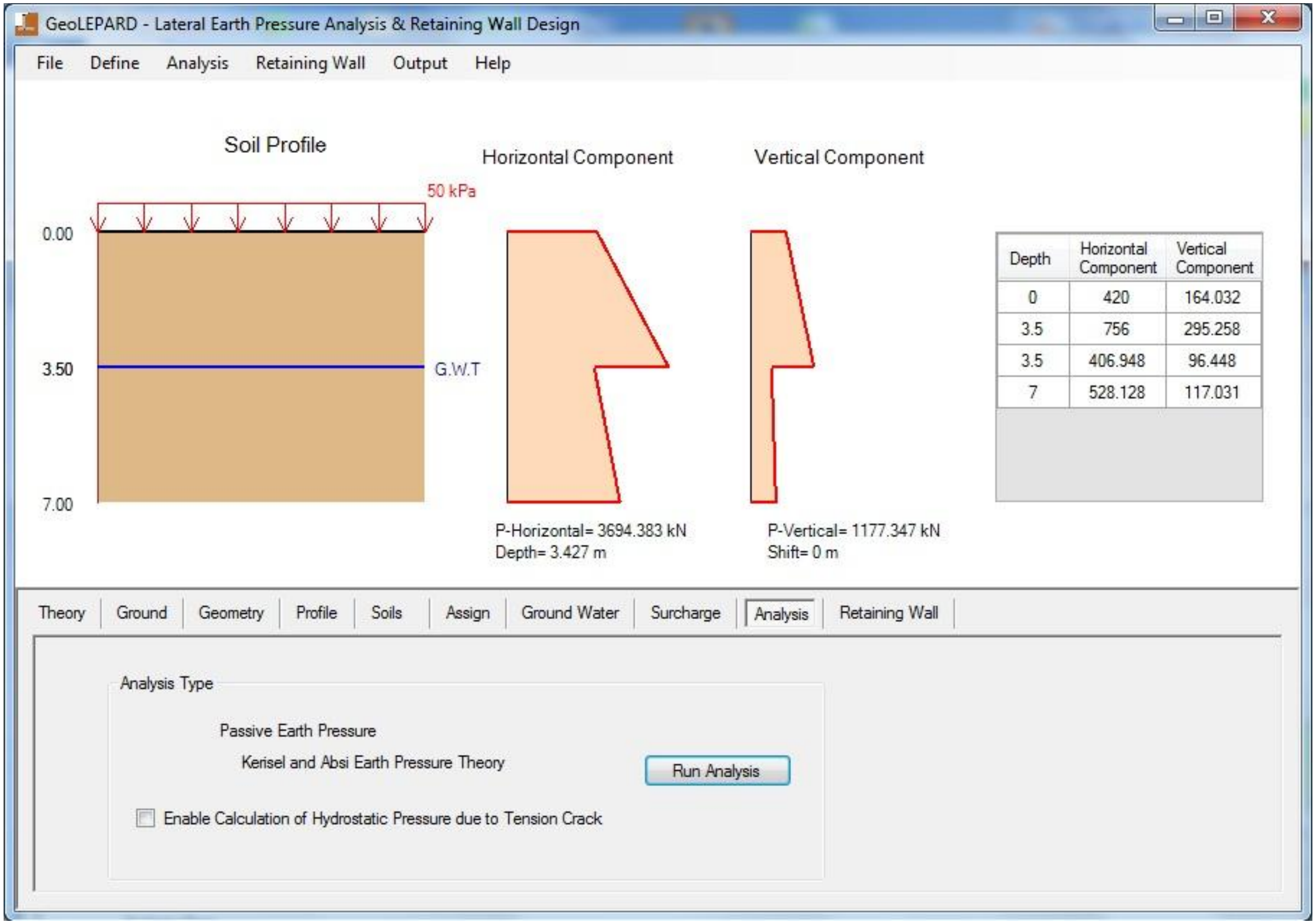
Shift of Centroid:

$$\bar{X} = 0$$

II. Results from the developed software



(a) Active Earth Pressure



(b) Passive Earth Pressure

Figure: E6.4.5: Graphical and tabular Solution from the developed software

7. Plot the active and passive earth pressure diagrams and compute the horizontal and vertical force components of the resultant force together with its point of application for the wall system shown in Fig. E6.1.5 using the following Earth Pressure Theories :

   a. Rankine (Bell)
   b. Coulomb's
   c. Kerisel and Absi

Assume: $\delta = \frac{2}{3}\emptyset$

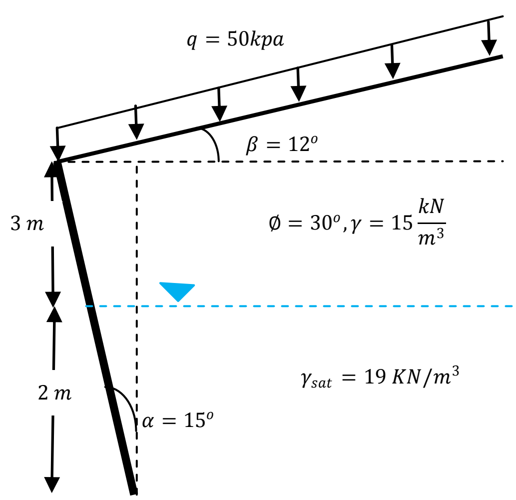


Figure: E7.1.5

a. **Rankine (Bell)**

I. **Results from Hand/Spread Sheet Calculations**

$k_a = 0.357$ from Eqn. 2.26

$k_p = 2.678$ from Eqn. 2.27

We compute the Rankine Earth pressure about a fictitious straight line drawn from the heel towards the ground. Due to this we'll have an additional height of:
$\boldsymbol{H * \tan\alpha \tan\beta = 0.2848\ m}$, as shown in figure E7.2.5.

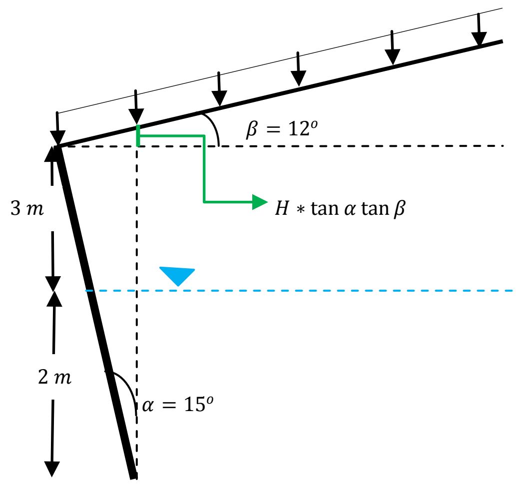


Figure: E7.2.5

Table E7.1.5: Computation of Active Earth Pressure

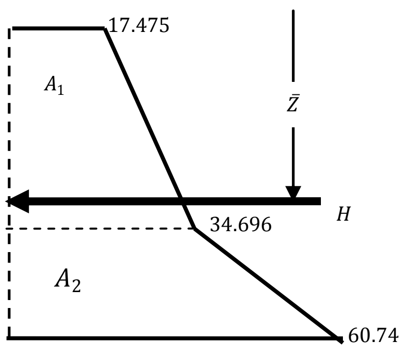
Depth (m), From top to Bottom	Coefficient of Earth pressure (Ka)	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	0.357	50	15			9.81		17.866	0.000	17.475	3.715
3.285	0.357	50	15			9.81		35.471	0.000	34.696	7.375
5.285	0.357	50		19	3	9.81	9.19	42.039	19.620	60.740	8.740

Table E7.2.5: Computation of Passive Earth Pressure

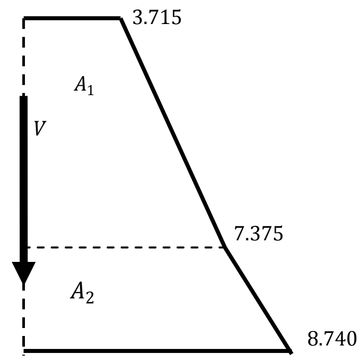
Depth (m), From top to Bottom	Coefficient of Earth pressure (Kp)	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	2.678	50	15			9.81		133.883	0.000	130.958	27.836
3.285	2.678	50	15			9.81		265.816	0.000	260.008	55.266
5.285	2.678	50		19	3	9.81	9.19	315.032	19.620	327.768	65.499

- **<u>Lateral Earth pressure diagram:</u>**

<u>Active Case</u>



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{181.122\ kN/m}$$

<u>Depth of Centroid:</u>

$$\bar{Z} = \frac{\sum_{i=1}^{n} \mathrm{A_i\, Z_i}}{\sum_{i=1}^{n} \mathrm{A_i}}$$

Horizontal Component: $\bar{Z} = \frac{573.817}{181.122} = \mathbf{3.168\ m}$ from the top of the fictitious line

$\bar{Z} = \mathbf{3.168\ m - 0.2848\ m} = \mathbf{2.883\ m}$ from the top of the retaining wall

Total Vertical Force:

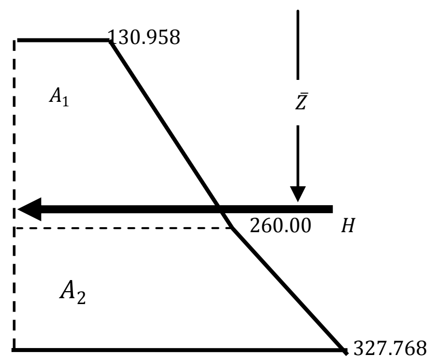
$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{34.328\ kN/m}$$

<u>Shift of Centroid:</u>

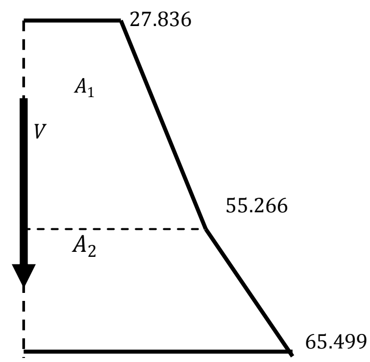
$\bar{X} = \mathrm{H} * \tan \alpha$

$\bar{X} = 5 * \tan 15 = \boldsymbol{1.34\ m}$

Passive Case



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{1229.89\ kN/m}$$

Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{n} \mathrm{A_i}\,\mathrm{Z_i}}{\sum_{i=1}^{n} \mathrm{A_i}}$$

Horizontal Component: $\bar{Z} = \frac{651.963}{200.57} = \mathbf{3.018\ m}$ from top of the fictitious line

$\bar{Z} = \mathbf{3.018\ m - 0.2848\ m\ = 2.733\ m}$ From top of the retaining wall

Total Vertical Force:

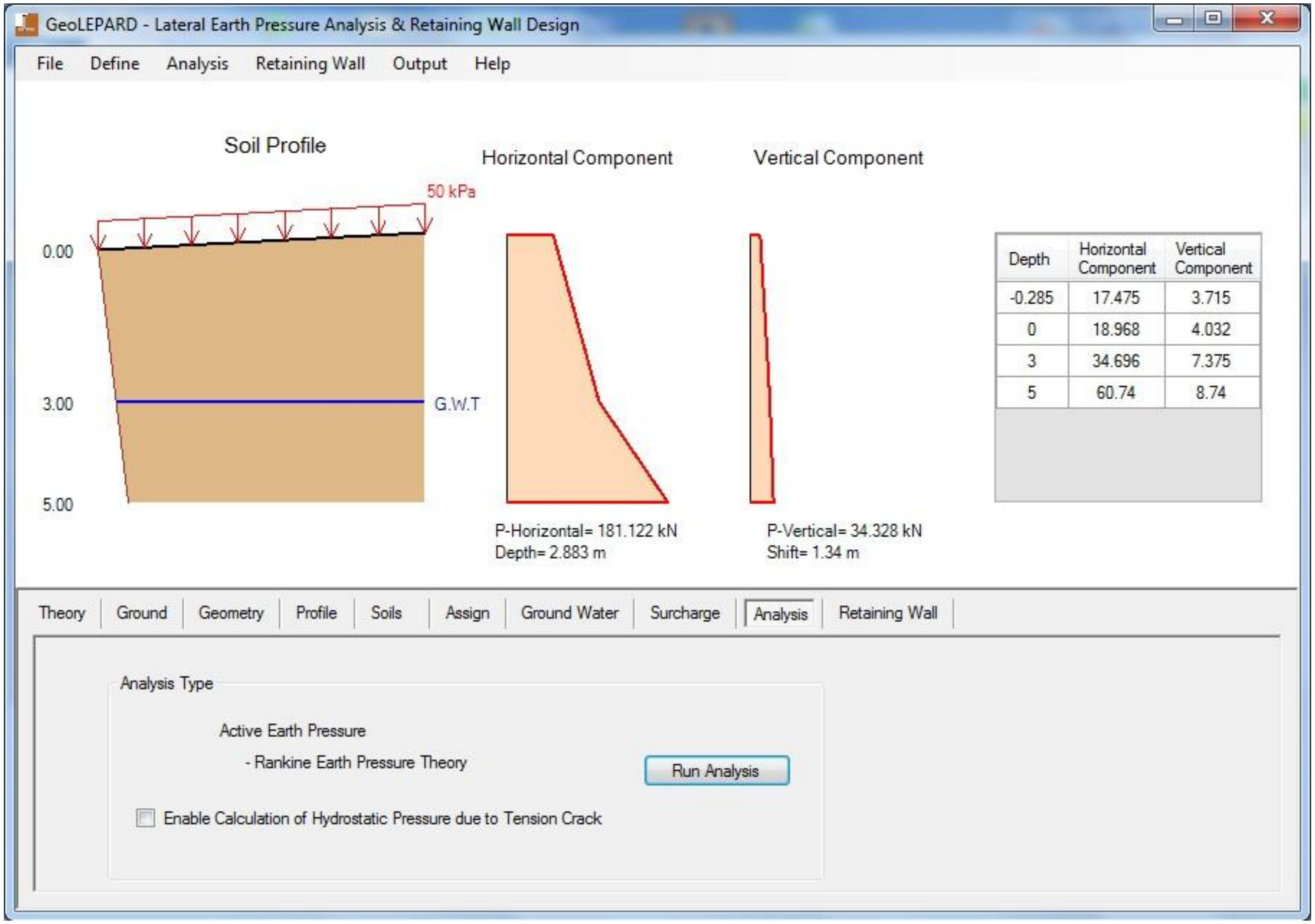
$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{257.251\ kN/m}$$

Shift of Centroid:

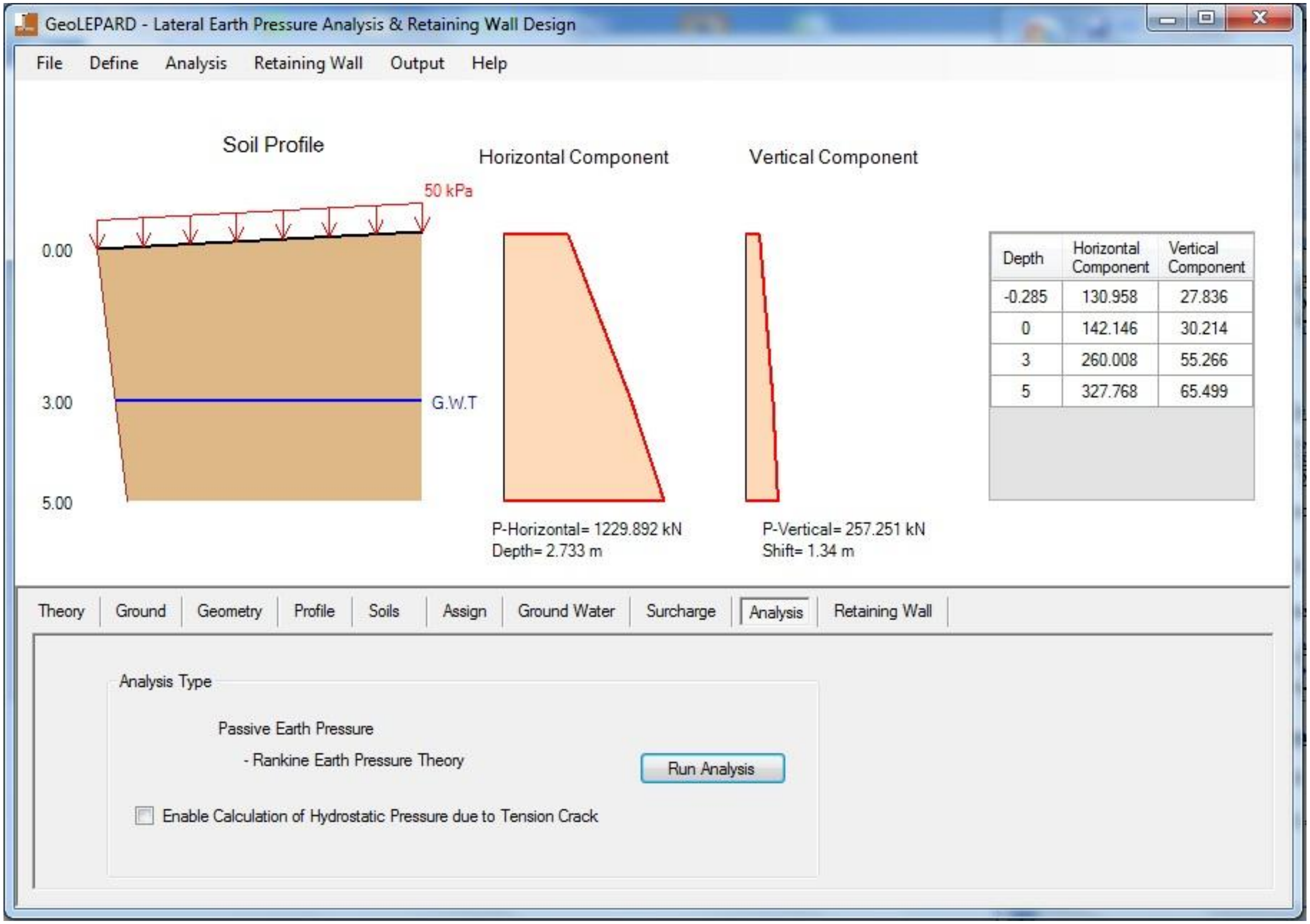
$$\bar{X} = \mathrm{H} * \tan\alpha$$

$$\bar{X} = 5 * \tan 15 = \mathbf{1.34}\ \boldsymbol{m}$$

II. Results from the developed software



(b) Active Earth Pressure



(c) Passive Earth Pressure

Figure: E7.3.5: Graphical and tabular Solution from the developed software

b. Coulomb

I. Results from Hand/Spread Sheet Calculations

$k_a = 0.516$ From Eqn. 2.49

$k_p = 6.755$ From Eqn. 2.61

Table E7.3.5: Computation of Active Earth Pressure

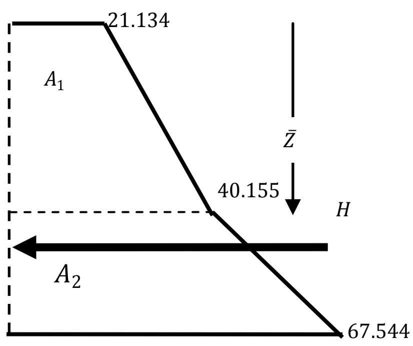
Depth (m), From top to Bottom	Coefficient of Earth pressure (Ka)	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	0.516	50	15			9.81		25.800	0.000	21.134	14.798
3	0.516	50	15			9.81		49.020	0.000	40.155	28.117
5	0.516	50		19	2	9.81	9.19	58.504	19.620	67.544	33.557

Table E7.4.5: Computation of Passive Earth Pressure

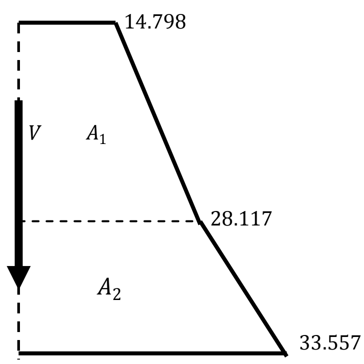
Depth (m), From top to Bottom	Coefficient of Earth pressure (Kp)	Uniform Surcharge	Bulk unit weight	Saturated unit weight	Depth below GWT (m)	Water unit weight	Effective Unit weight	Total Lateral Earth pressure	Hydrostatic Pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	6.755	50	15			9.81		337.75	0	336.465	29.437
3	6.755	50	15			9.81		641.725	0	639.283	55.93
5	6.755	50		19	2	9.81	9.19	765.8819	19.62	782.587	66.751

- **Lateral Earth pressure diagram:**

Active Case



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{199.63\ kN/m}$$

Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{n} \mathrm{A_i\ Z_i}}{\sum_{i=1}^{n} \mathrm{A_i}}$$

Horizontal Component: $\bar{Z} = \frac{592.07}{199.63} = \mathbf{2.966\ m}$

Total Vertical Force:

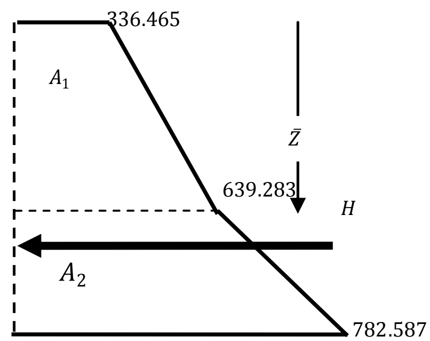
$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{126.046\ kN/m}$$

Shift of Centroid:

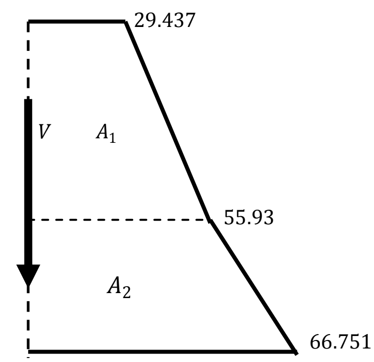
$$\bar{X} = \bar{Z} * \tan \alpha$$

$$\bar{X} = 2.817 * \tan 15 = \mathbf{0.755\ \mathit{m}}$$

Passive Case



Horizontal Component



Vertical Component

Total Horizontal Force:

$$H = \sum_{1}^{n} \mathrm{A_i} = \mathbf{2885.49\ kN/m}$$

Depth of Centroid:

$$\bar{Z} = \frac{\sum_{i=1}^{n} \mathrm{A_i\ Z_i}}{\sum_{i=1}^{n} \mathrm{A_i}}$$

Horizontal Component: $\bar{Z} = \frac{6723.11}{2376.173} = \mathbf{2.827\ m}$

Total Vertical Force:

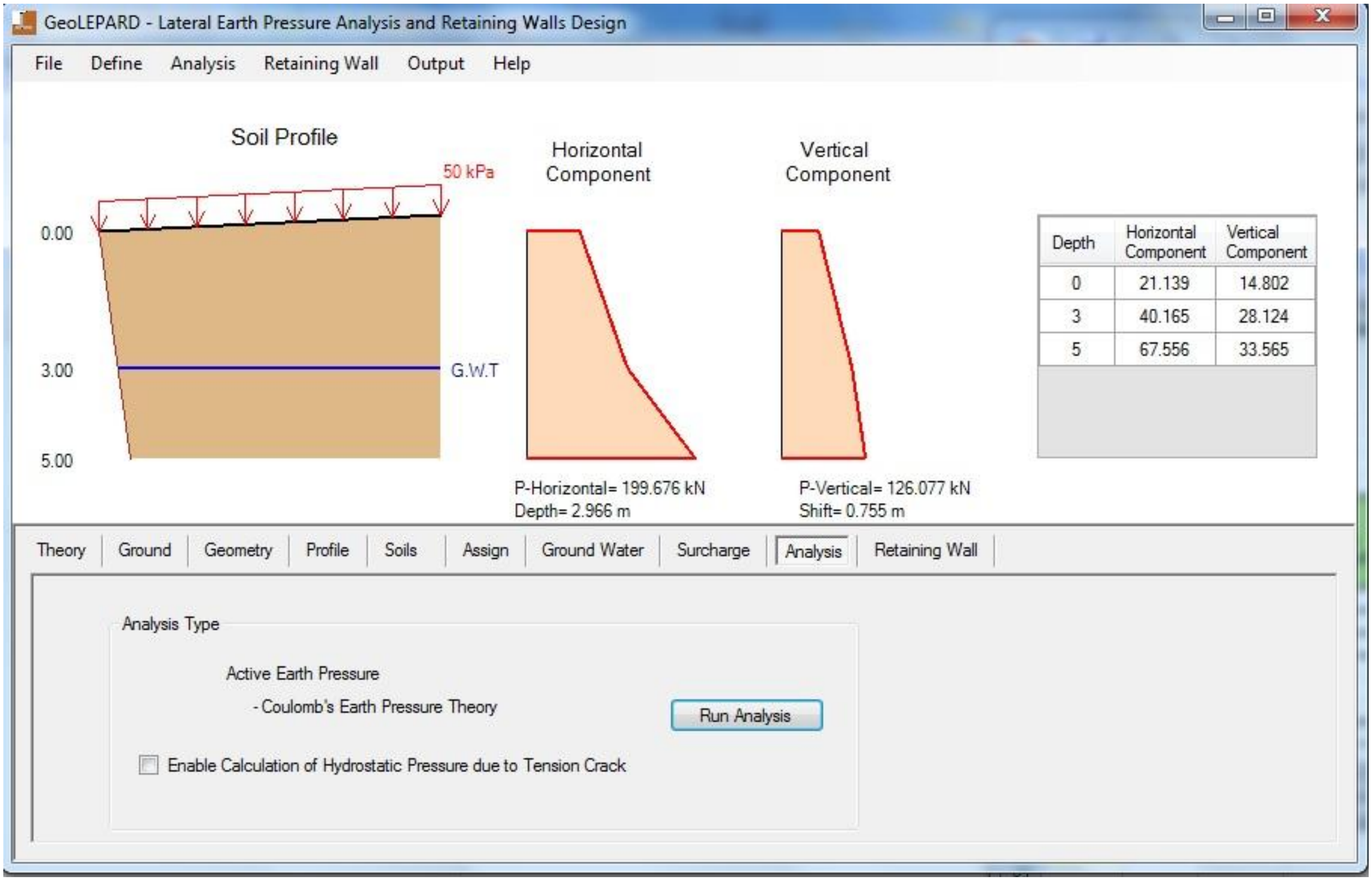
$$V = \sum_{1}^{n} \mathrm{A_i} = \mathbf{250.7\ kN/m}$$

Shift of Centroid:

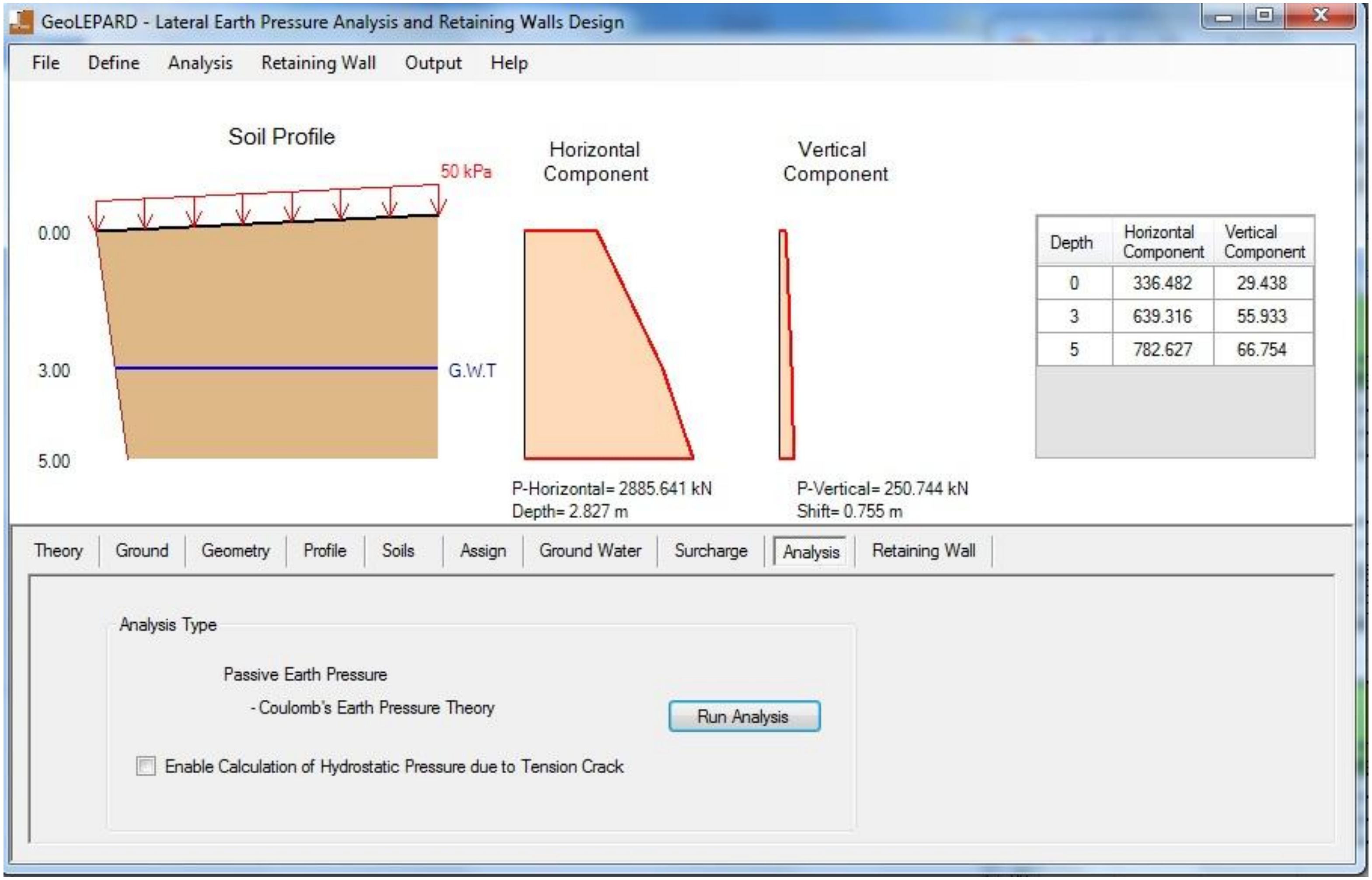
$$\bar{X} = \bar{Z} * \tan \alpha$$

$$\bar{X} = 2.817 * \tan 15 = \mathbf{0.755}\ \boldsymbol{m}$$

II. Results from the developed software



(a) Active Earth Pressure



(b) Passive Earth Pressure

Figure: E7.4.5: Graphical and tabular Solution from the developed software

c. **Kerisel & Absi**

In this theory the software computes only for vertical wall and horizontal ground.

8. Consider the soil profile in Example-6 above to compute the active pressure at the back of the wall shown below and analyze it for external stability. For the front face soil; $\emptyset = 15^0, C = 15kpa\ and\ \gamma = 15kN/m^3$, $\gamma_{sat} = 18kN/m^3$ and for the foundation soil : $\emptyset = 35^0, C = 20kpa\ , \gamma = 16kN/m^3$, $\gamma_{sat} = 19kN/m^3$,
Use following Earth Pressure Theories for analysis:

   a. Rankine (Bell)
   b. Coulomb's
   c. Kerisel and Absi

Assume: $\delta = \frac{2}{3}\emptyset, and\ c_w = 0.5c,\ \gamma_{wall} = 23kN/m^3$ $FS_{sl} = 2,\ FS_{ot} = 2\ , FS_{BC} = 2$ and the front face passive pressure reduction factor to be 2/3

Use Meyerhof's the General Bearing Capacity Equation to compute the ultimate bearing capacity.

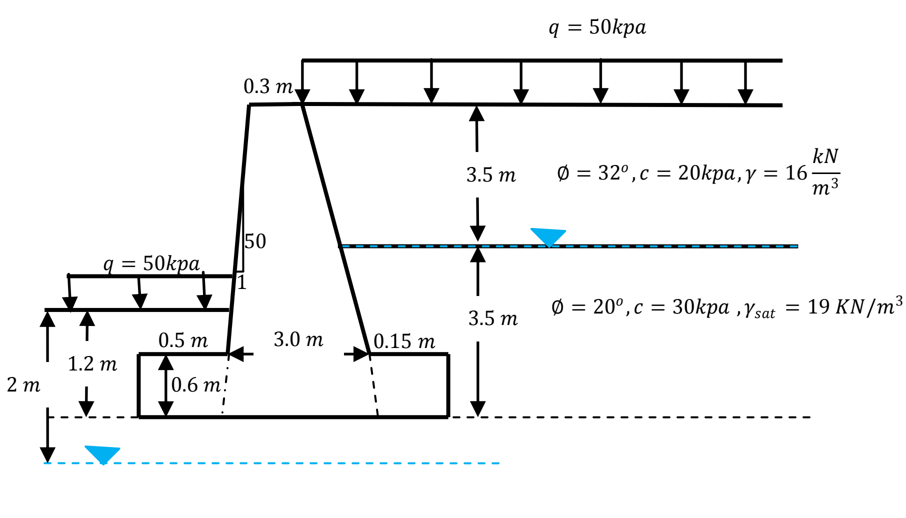


Figure E8.1.5

a. <u>Rankine (Bell)</u>

I. Results from Hand/Spread Sheet Calculations

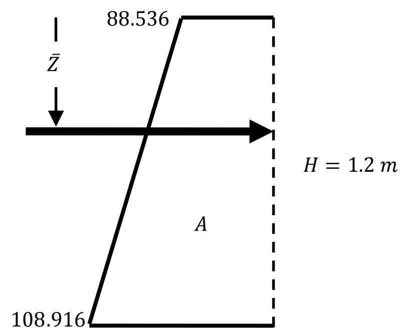
Coefficient of earth pressure for the front face soil:

$k_p = 1.698$ From Eqn. 2.19b:

Table E8.1.5: Computation of Passive pressure for the front face soil

Depth (m)	Kp	Reduced Kp	Total Lateral Earth pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	1.698	1.132	88.536	88.536	0
1.2	1.698	1.132	108.916	108.916	0

<u>Passive Case</u>



Horizontal Component

Total Horizontal Force:

$H = \text{Area} = 118.471\text{kN/m}$

<u>Depth of Centroid:</u>

$\bar{Z} = 0.62\ \text{m}$

Force diagram:

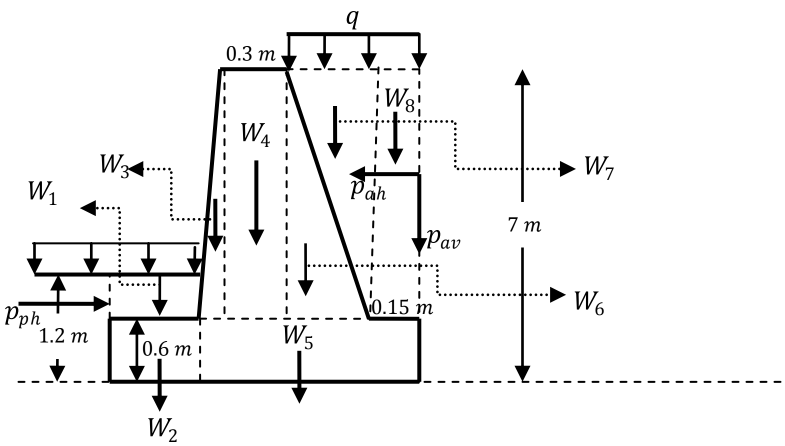


Figure E8.2.5: Force diagram

Computation of Resisting Moment

Table E8.2.5: Resisting Moment Computation

Part	Unit Weight (KN/m³)	Force (KN/m)	Moment arm from the toe (m)	Moment about the toe (KN-m/m)
W_1	15	30.154	0.254	7.666
W_2	23	6.900	0.250	1.725
W_3	23	9.421	0.585	5.514
W_4	23	44.160	0.778	34.356
W_5	23	43.470	2.075	90.200
W_6	23	189.299	1.785	337.962
W_7	soil part	136.732	2.643	361.336
	surcharge	128.600	2.214	284.720
W_8		24.165	3.575	86.390
P_{av}		0.000	3.650	0.000
	ΣF_v =	**612.901**	**ΣM_r =**	**1209.871**

Computation of overturning Moment

Force(KN/m)	Moment arm from the toe (m)	Moment about the toe (Mo) (KN-m/m)
133.54[a]	1.569	209.524

a-value is taken from Example-6

Stability against overturning

$$\frac{\text{Resisting Moment}}{\text{Overturning Moment}} \geq FS_{ot}$$

$$\frac{1209.871}{209.524} = 5.774 > 2 \ \boldsymbol{OK}$$

Stability against Sliding

$$\boldsymbol{F_r} = \mathbf{F_v} \tan\delta + c_w B + P_{ph}$$

$$\boldsymbol{F_r} = \mathbf{338.462}\ \boldsymbol{kN/m}$$

$Driving\ force(\boldsymbol{F_d}) = \boldsymbol{p_{ah}} = \mathbf{133.54}\boldsymbol{KN/m}$ **:** From Example 6 above

$$\frac{\boldsymbol{F_r}}{\boldsymbol{F_d}} = \frac{\mathbf{338.462}}{\mathbf{133.54}} = 2.535 > 2 \quad \boldsymbol{OK}$$

Check for Toe over-stress

$$\bar{X} = \frac{Resisting\ Moment - Overturning\ Moment}{\boldsymbol{F_v}}$$

$$\bar{X} = \frac{1209.871 - 209.524}{612.901} = 1.632\ m$$

$$e = \frac{B}{2} - \bar{X}$$

$$e = \frac{3.65}{2} - 1.632$$

$$= 0.193m < \frac{B}{6} = 0.608\ m \ \boldsymbol{OK}$$

Stability against Bearing Capacity failure

$$q_{min} = \frac{v}{B}\left[1 - \frac{6e}{B}\right]$$

$$q_{min} = 114.686\ kN/m^2$$

$$q_{max} = \frac{v}{B}\left[1 + \frac{6e}{B}\right]$$

$$q_{max} = 221.15\ kN/m^2$$

Bearing Capacity Computation Using Meyerhof's General Bearing Capacity Equation

Using Eqn (2.80):

C-terms		q-terms		B-terms	
C	20	q	68	B	3.65
N_c	20.721	N_q	10.662	γ	10.683
F_{cs}	1	F_{qs}	1	$N\gamma$	10.876
F_{cd}	1.113	F_{qd}	1.102	$F\gamma s$	1
F_{ci}	0.746	F_{qi}	0.746	$F\gamma d$	1
				$F\gamma i$	0.508

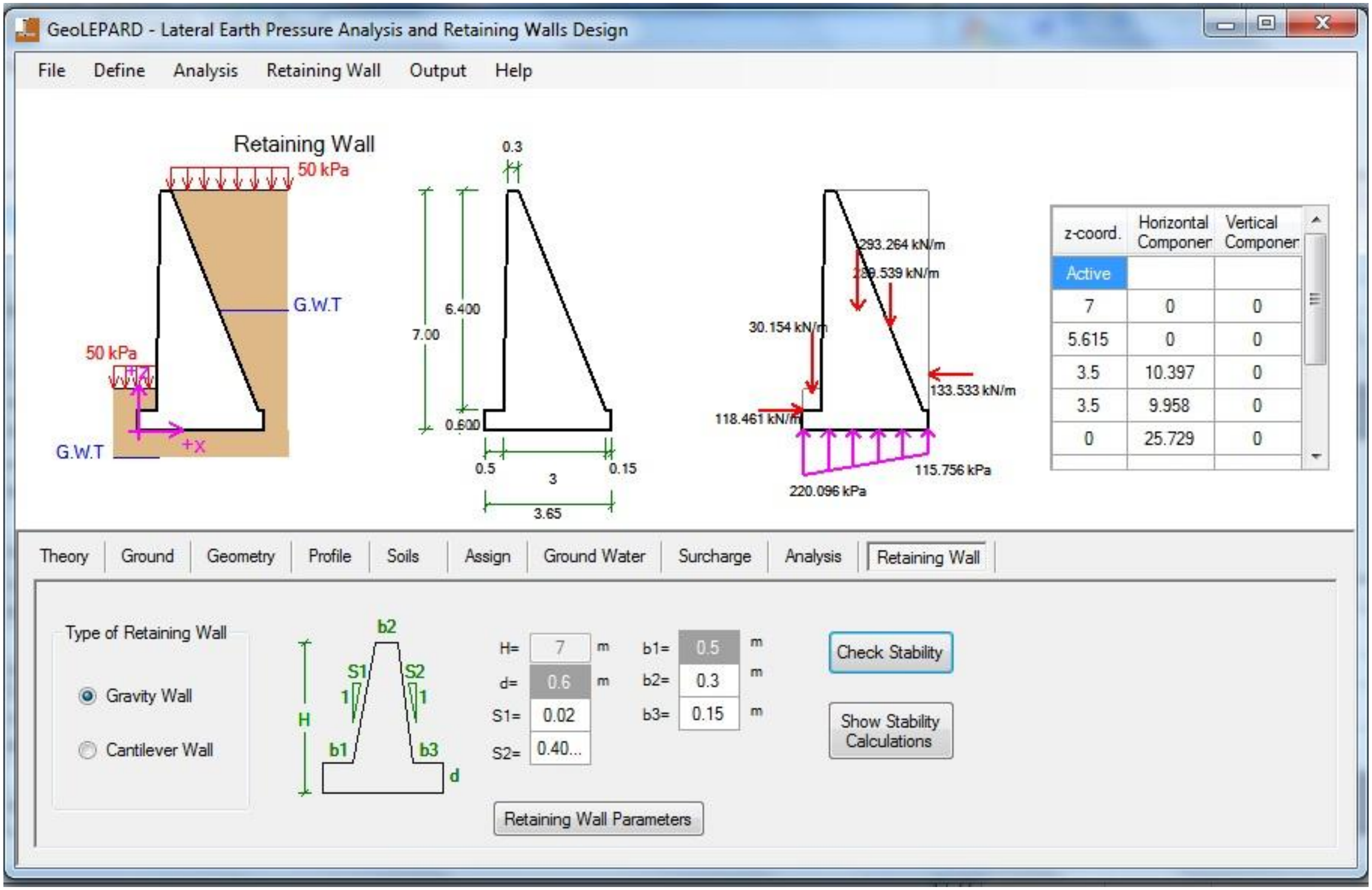
$$q_u = 20 * 20.721 * 1 * 1.113 * 0.746 + 68 * 10.662 * 1 * 1.102 * 0.746 + 3.65 * 10.683 * 10.876 * 1 * 1 * 0.508$$

$$= \mathbf{1047.397}\ Kpa$$

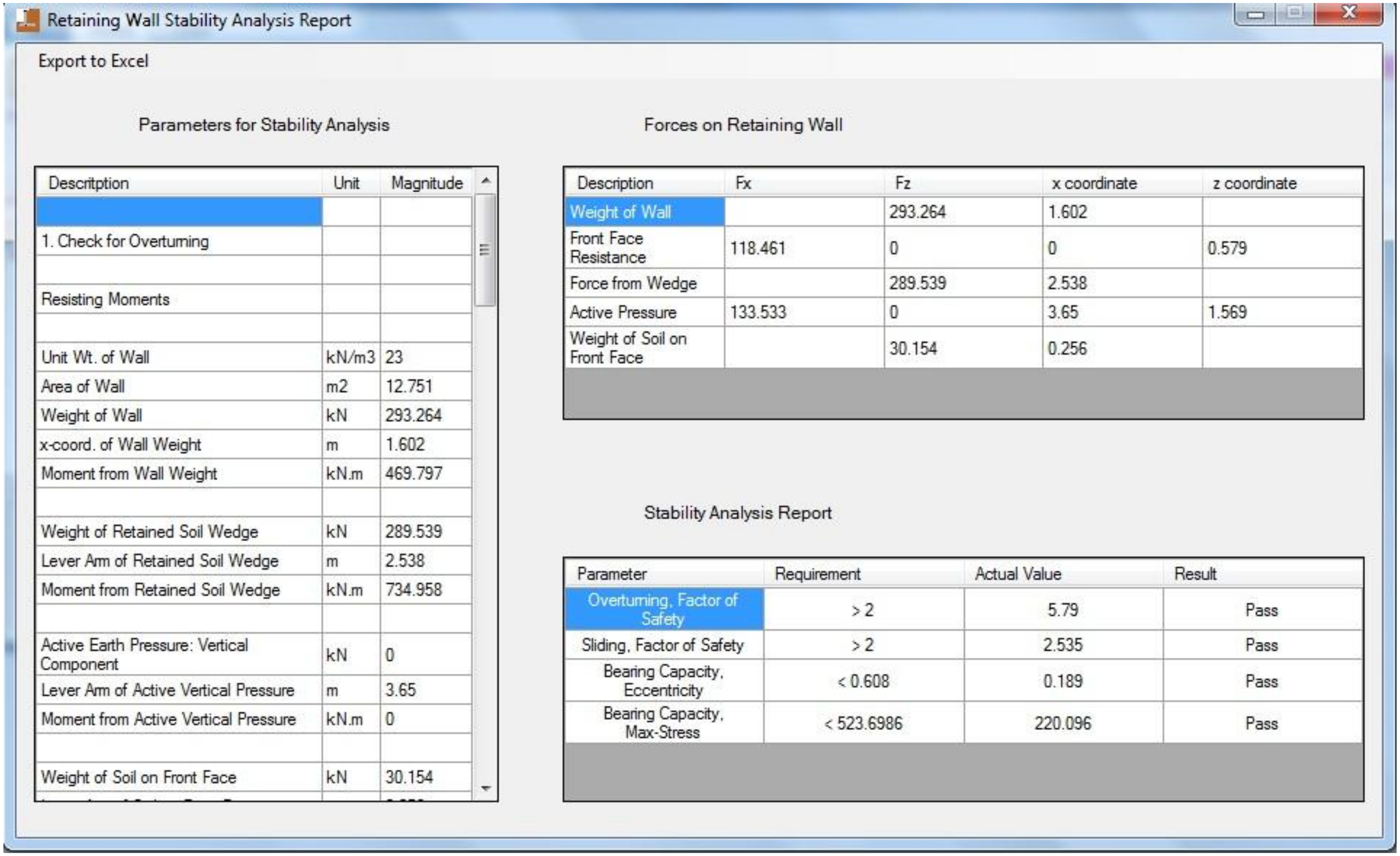
$$q_{all} = \frac{1047.397}{2} = 523.699 Kpa$$

$q_{all} = 523.699 > q_{max} = 221.15$ **OK**

II. Results from the developed Software



(a) Graphical Output


Retaining Wall Stability Analysis Report

Export to Excel

Parameters for Stability Analysis

Descritption	Unit	Magnitude
1. Check for Overturning		
Resisting Moments		
Unit Wt. of Wall	kN/m3	23
Area of Wall	m2	12.751
Weight of Wall	kN	293.264
x-coord. of Wall Weight	m	1.602
Moment from Wall Weight	kN.m	469.797
Weight of Retained Soil Wedge	kN	289.539
Lever Arm of Retained Soil Wedge	m	2.538
Moment from Retained Soil Wedge	kN.m	734.958
Active Earth Pressure: Vertical Component	kN	0
Lever Arm of Active Vertical Pressure	m	3.65
Moment from Active Vertical Pressure	kN.m	0
Weight of Soil on Front Face	kN	30.154

Forces on Retaining Wall

Description	Fx	Fz	x coordinate	z coordinate
Weight of Wall		293.264	1.602	
Front Face Resistance	118.461	0	0	0.579
Force from Wedge		289.539	2.538	
Active Pressure	133.533	0	3.65	1.569
Weight of Soil on Front Face		30.154	0.256	

Stability Analysis Report

Parameter	Requirement	Actual Value	Result
Overturning, Factor of Safety	> 2	5.79	Pass
Sliding, Factor of Safety	> 2	2.535	Pass
Bearing Capacity, Eccentricity	< 0.608	0.189	Pass
Bearing Capacity, Max-Stress	< 523.6986	220.096	Pass

(b) Tabular Output

Figure: E8.3.5: Graphical and tabular Solution from the developed software

b. Coulomb's Earth Pressure Theory

I. Results from Hand/Spread Sheet Calculations

As the length of the heel is only 15cm; we could have omitted the presence of the heel in the calculation of the lateral earth pressure. However, as the soil profile involves cohesion, for which only horizontal ground and vertical wall is considered, we will assume a fictitious vertical plane passing through the heel.

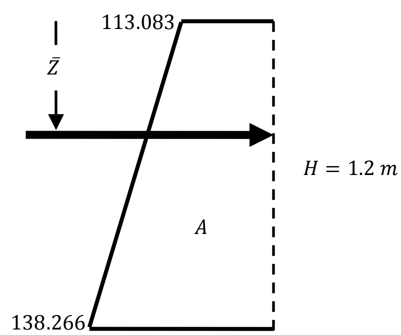
Computation of Resisting Moment

Table E8.5.5: Resisting Moment Computation

Part	Unit Weight (KN/m³)	Force (KN/m)	Moment arm from the toe (m)	Moment about the toe (KN-m/m)
W_1	15	30.154	0.254	7.666
W_2	23	6.900	0.250	1.725
W_3	23	9.421	0.585	5.514
W_4	23	44.160	0.778	34.356
W_5	23	43.470	2.075	90.200
W_6	23	189.299	1.785	337.962
W_7		136.732	2.643	361.336
		128.600	2.214	284.720
W_8		24.165	3.575	86.390
P_{av}		4.538	3.650	16.564
	ΣF_v=	**617.440**	**ΣM_r=**	**1226.438**

Table : Computation of Passive pressure for the front face soil

Depth (m)	Kp	Reduced Kp	Total Lateral Earth pressure	Lateral Earth pressure (Horizontal)	Lateral Earth pressure (Vertical)
0	2.131	1.42	114.83	113.083	19.94
1.2	2.131	1.42	140.399	138.266	24.380



Horizontal Component

Total Horizontal Force:

$H = \text{Area} = 150.81\text{kN/m}$

Depth of Centroid:

$\bar{Z} = 0.62$ m

Computation of overturning Moment

Force(KN/m)	Moment arm from the toe (m)	Moment about the toe (Mo) (KN-m/m)
78.428	1.169	91.682

Stability against overturning

$$\frac{\text{Resisting Moment}}{\text{Overturning Moment}} \geq FS_{ot}$$

$$\frac{1226.438}{91.682} = 13.377 > 2\ \boldsymbol{OK}$$

Stability against Sliding

$$\boldsymbol{F_r} = \mathbf{F_v} \tan\delta + c_w B + P_{ph}$$

$$\boldsymbol{F_r} = \mathbf{372.159}\boldsymbol{KN/m}$$

$$Driving\ force(\boldsymbol{F_d}) = \boldsymbol{p_{ah}} = \mathbf{78.428}\boldsymbol{kN/m}$$

$$\frac{\boldsymbol{F_r}}{\boldsymbol{F_d}} = \frac{\mathbf{372.159}}{\mathbf{78.428}} = 4.745 > 2 \quad \boldsymbol{OK}$$

Check for Toe over-stress

$$\bar{X} = \frac{Resisting\ Moment - Overturning\ Moment}{\boldsymbol{F_v}}$$

$$\bar{X} = \frac{1226.438 - 91.682}{617.440} = 1.838m$$

$$e = \frac{B}{2} - \bar{X}$$

$$e = \left|\frac{3.65}{2} - 1.838\right|$$

$$= 0.013\ m > \frac{B}{6} = 0.608\ m\ \boldsymbol{OK}$$

Stability against Bearing Capacity failure

$$q_{min} = \frac{v}{B}\left[1 - \frac{6e}{B}\right]$$

$$q_{min} = 165.591\ kN/m^2$$

$$q_{max} = \frac{v}{B}\left[1 + \frac{6e}{B}\right]$$

$$q_{max} = 172.732\ kN/m^2$$

Bearing Capacity Computation Using Meyerhof's General Bearing Capacity Equation

Using Eqn (2.80):

C-terms		q-terms		B-terms	
C	20	q	68	B	3.65
N_c	20.721	N_q	10.662	γ	10.683
F_{cs}	1	F_{qs}	1	$N\gamma$	10.876
F_{cd}	1.113	F_{qd}	1.102	$F\gamma s$	1
F_{ci}	0.846	F_{qi}	0.846	$F\gamma d$	1
				$F\gamma i$	0.710

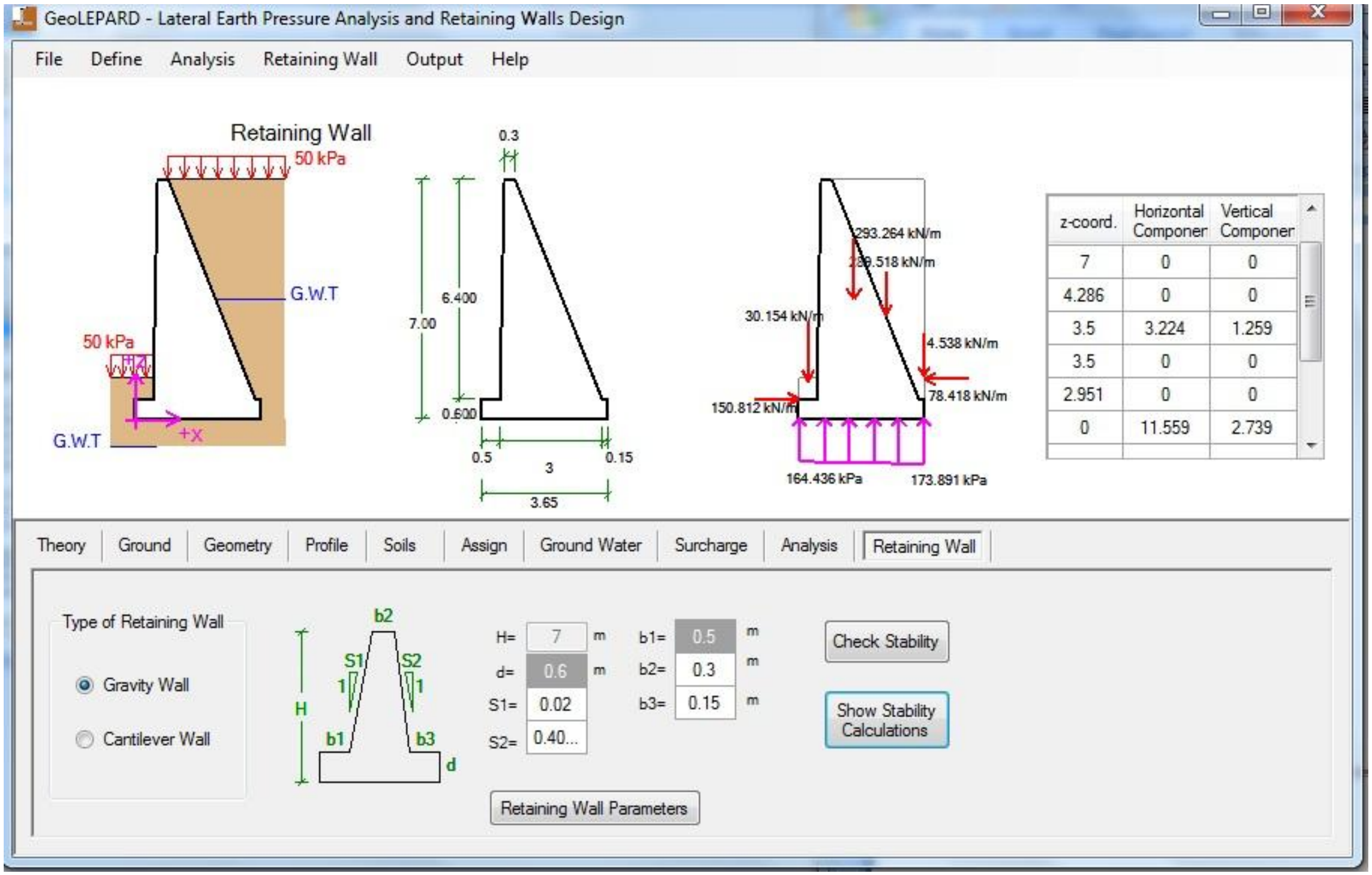
$$q_u = 20 * 20.721 * 1 * 1.113 * 0.846 + 68 * 10.662 * 1 * 1.102 * 0.846 + 3.65 * 10.683 * 10.876 * 1 * 1 * 0.710$$

$$= \mathbf{1216.391}\ Kpa$$

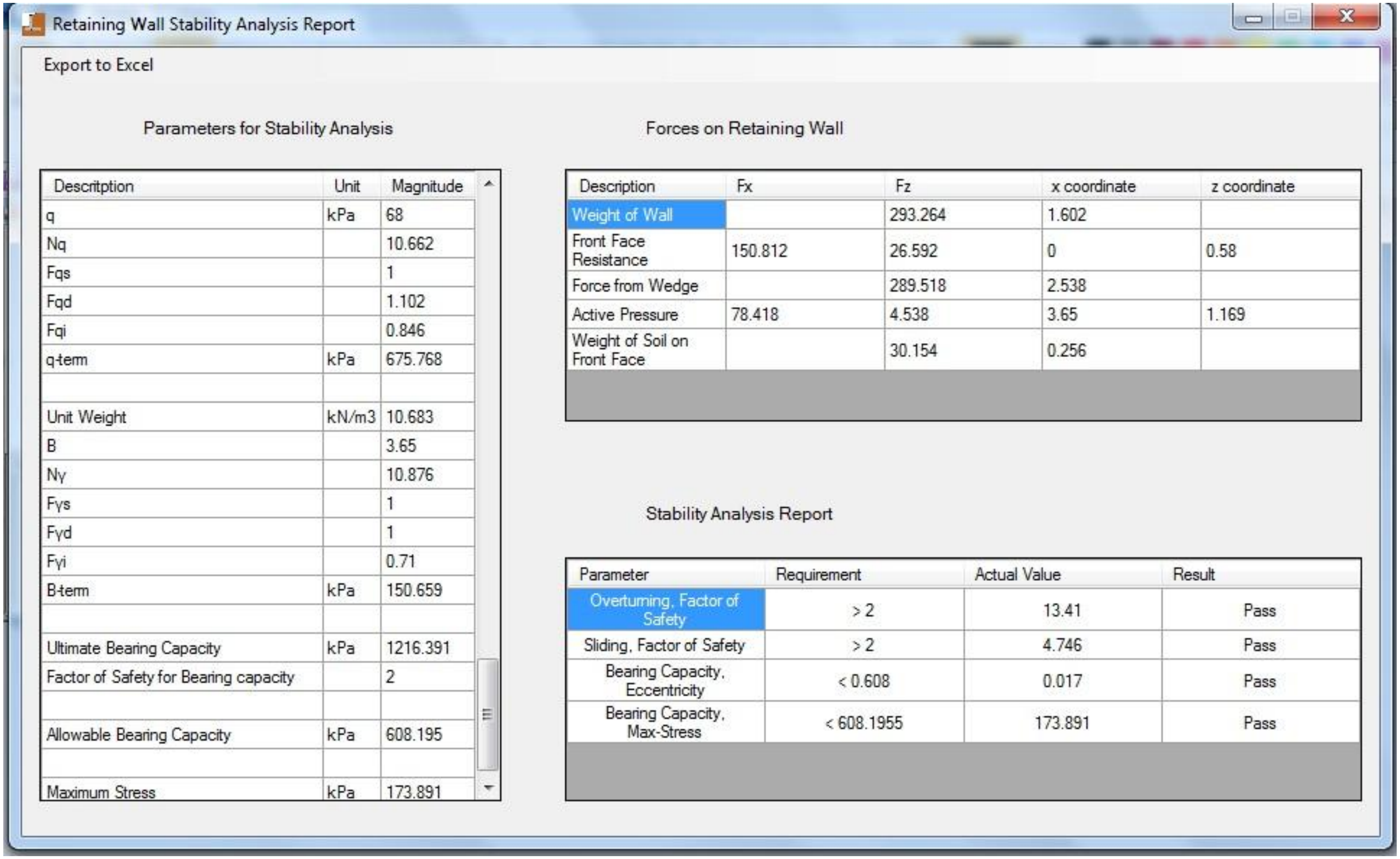
$$q_{all} = \frac{1216.391}{2} = 608Kpa$$

$q_{all} = 608 >> q_{max} = 172.732$ **OK**

II. Results from the developed Software



(a) Graphical Output



Retaining Wall Stability Analysis Report

Export to Excel

Parameters for Stability Analysis

Descritption	Unit	Magnitude
q	kPa	68
Nq		10.662
Fqs		1
Fqd		1.102
Fqi		0.846
q-term	kPa	675.768
Unit Weight	kN/m3	10.683
B		3.65
Nγ		10.876
Fγs		1
Fγd		1
Fγi		0.71
B-term	kPa	150.659
Ultimate Bearing Capacity	kPa	1216.391
Factor of Safety for Bearing capacity		2
Allowable Bearing Capacity	kPa	608.195
Maximum Stress	kPa	173.891

Forces on Retaining Wall

Description	Fx	Fz	x coordinate	z coordinate
Weight of Wall		293.264	1.602	
Front Face Resistance	150.812	26.592	0	0.58
Force from Wedge		289.518	2.538	
Active Pressure	78.418	4.538	3.65	1.169
Weight of Soil on Front Face		30.154	0.256	

Stability Analysis Report

Parameter	Requirement	Actual Value	Result
Overturning, Factor of Safety	> 2	13.41	Pass
Sliding, Factor of Safety	> 2	4.746	Pass
Bearing Capacity, Eccentricity	< 0.608	0.017	Pass
Bearing Capacity, Max-Stress	< 608.1955	173.891	Pass

(b) Tabular Output

Figure: E8.4.5: Graphical and tabular Solution from the developed software

c. Kerisel and Absi Earth Pressure Theory

The theory applies to only vertical wall and horizontal ground. Hence to compute the lateral pressure we will assume a fictitious vertical plane drawn from the heel towards the ground.

The active side (back face of wall) has already been computed in Example -6.

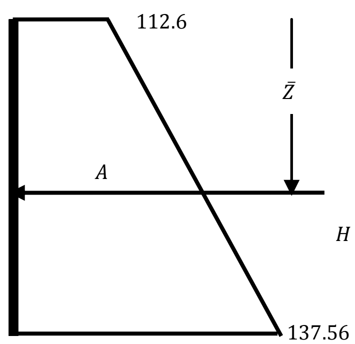
Coefficients of earth pressure for soil at the front face:

$k_p = 2.08, k_{pc} = 3.533$

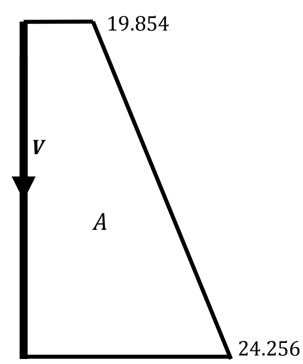
Table 8.6.8: Computation of Passive Earth Pressure (Front Face)

Depth (m), From top to Bottom	Coefficient of Earth pressure (Kp)	Reduced Coefficient of Earth pressure (K_{pR})	Uniform Surcharge (KN/m^2)	Cohesion (KN/m^2)	Bulk unit weight (KN/m^3)	Lateral Earth pressure (Horizontal) (KN/m^2)/m	Lateral Earth pressure (Vertical) (KN/m^2)/m
0	2.08	1.387	50	15	15	112.6	19.854
1.2	2.08	1.387	50	15	15	137.56	24.256

- **Lateral Earth pressure diagram:**



Horizontal Component



Vertical Component

Total Horizontal Force:

$H = \mathrm{A} = \mathbf{150.81\ kN/m}$

Total Vertical Force:

$V = \mathrm{A} = \mathbf{26.59\ kN/m}$

Depth of Centroid: (horizontal component)

$\bar{Z} = \mathbf{0.62\ m}$

Shift of Centroid:

$\bar{X} = \mathbf{3.65}\ \boldsymbol{m}$

Computation of Resisting Moment

Using the force diagram in Fig: 8.2.5

Table E8.7.8: Resisting Moment Computation

Part	Unit Weight (KN/m³)	Force (KN/m)	Moment arm from the toe (m)	Moment about the toe (KN-m/m)
W_1	15	30.154	0.254	7.666
W_2	23	6.900	0.250	1.725
W_3	23	9.421	0.585	5.514
W_4	23	44.160	0.778	34.356
W_5	23	43.470	2.075	90.200
W_6	23	189.299	1.785	337.962
W_7	Soil	136.732	2.643	361.336
	Surcharge	128.600	2.214	284.720
W_8		24.165	3.575	86.390
P_{av}		3.995	3.650	14.582
	ΣF_v=	**616.896**	**ΣM_r=**	**1224.453**

Computation of overturning Moment

Force(KN/m)	Moment arm from the toe (m)	Moment about the toe (Mo) (KN-m/m)
76.582	1.139	87.220

Stability against overturning

$$\frac{\text{Resisting Moment}}{\text{Overturning Moment}} \geq FS_{ot}$$

$$\frac{1224.453}{87.220} = 14.039 > 2 \ \boldsymbol{OK}$$

Stability against Sliding

$$\boldsymbol{F_r} = \mathbf{F_v} \tan\delta + c_w B + P_{ph}$$

$$\boldsymbol{F_r} = \mathbf{371.282 kN/m}$$

$$Driving\ force(\boldsymbol{F_d}) = \boldsymbol{p_{ah}} = \mathbf{76.582\ kN/m}$$

$$\frac{\boldsymbol{F_r}}{\boldsymbol{F_d}} = \frac{\mathbf{371.282}}{\mathbf{76.582}} = 4.848 > 2 \quad \boldsymbol{OK}$$

Check for Toe over-stress

$$\bar{X} = \frac{Resisting\ Moment - Overturning\ Moment}{\boldsymbol{F_v}}$$

$$\bar{X} = \frac{1224.453 - 87.220}{616.896} = 1.843\ m$$

$$e = \frac{B}{2} - \bar{X}$$

$$e = \left|\frac{3.65}{2} - 1.843\right|$$

$$= |-0.018|\ m < \frac{B}{6} = 0.608\ m\ \boldsymbol{OK}$$

Stability against Bearing Capacity failure

$$q_{max} = \frac{v}{B}\left[1 + \frac{6e}{B}\right]$$

$$q_{max} = 174.146 kN/m^2$$

$$q_{min} = \frac{v}{B}\left[1 - \frac{6e}{B}\right]$$

$$q_{min} = 163.879\ kN/m^2$$

Bearing Capacity Computation Using Meyerhof's General Bearing Capacity Equation

Using Eqn (2.80):

C-terms		q-terms		B-terms	
C	20	q	68	B	3.65
N_c	20.721	N_q	10.662	γ	10.683
F_{cs}	1	F_{qs}	1	$N\gamma$	10.876
F_{cd}	1.113	F_{qd}	1.102	$F\gamma s$	1
F_{ci}	0.849	F_{qi}	0.849	$F\gamma d$	1
				$F\gamma i$	0.717

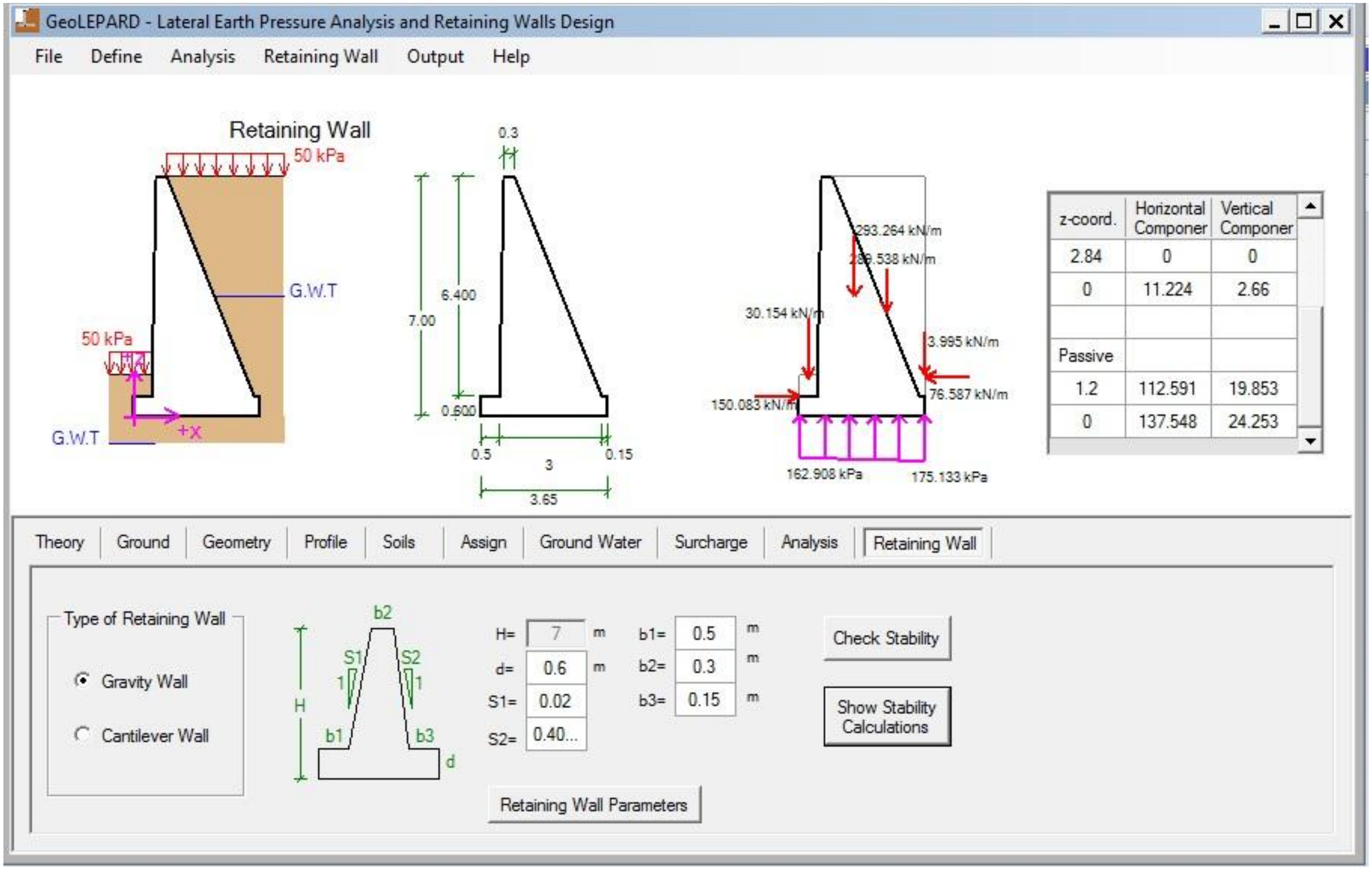
$$q_u = 20 * 20.721 * 1 * 1.113 * 0.849 + 68 * 10.662 * 1 * 1.102 * 0.849 + 3.65 * 10.683 * 10.876 * 1 * 1 * 0.717$$

$$= \mathbf{1221.92}\ Kpa$$

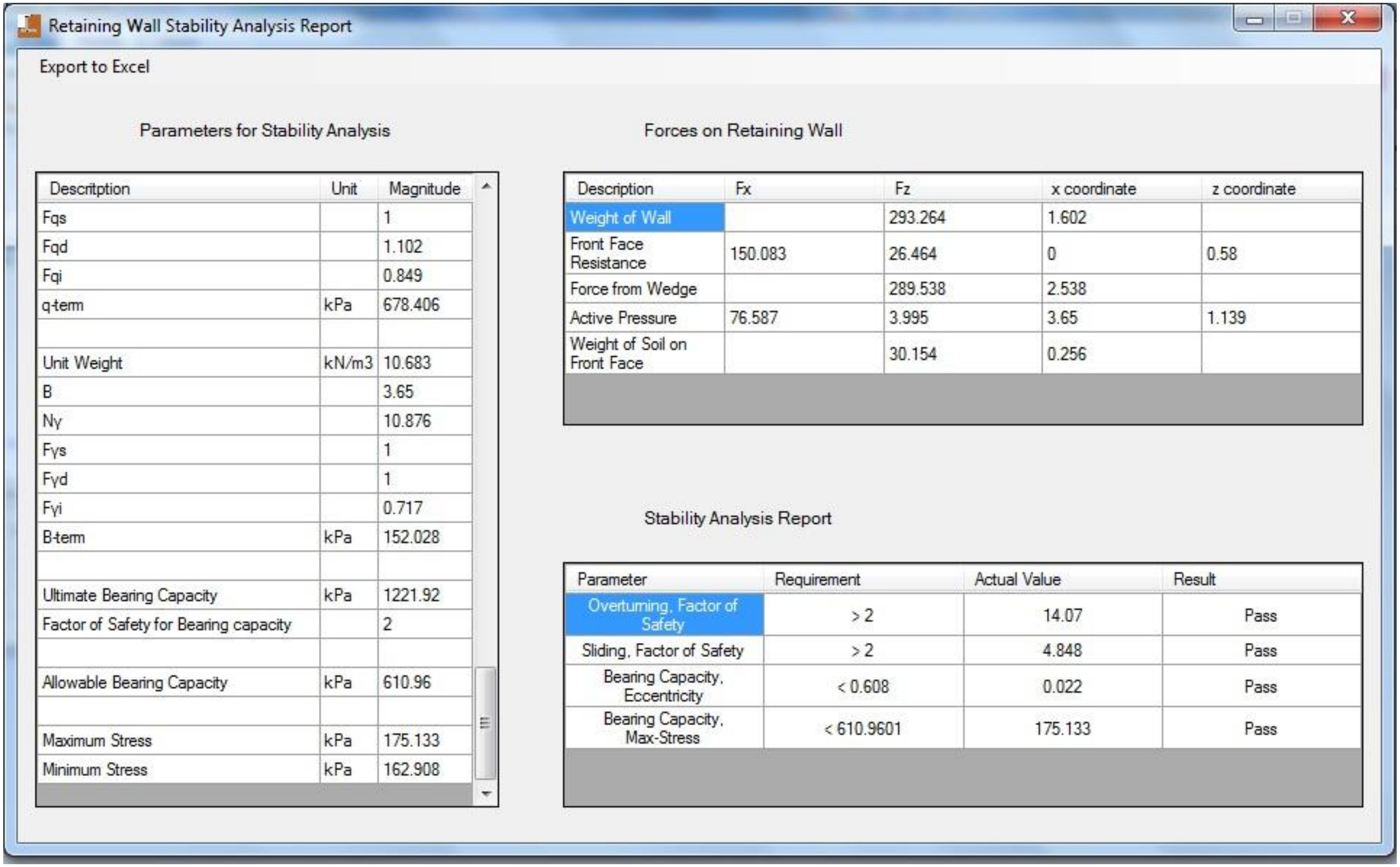
$$q_{all} = \frac{1221.92}{2} = 610.96Kpa$$

$q_{all} = 611 >> q_{max} = 175.132$ **OK**

Results from the developed Software



(a) Graphical Output



Retaining Wall Stability Analysis Report

Export to Excel

Parameters for Stability Analysis

Descritption	Unit	Magnitude
Fqs		1
Fqd		1.102
Fqi		0.849
q-term	kPa	678.406
Unit Weight	kN/m3	10.683
B		3.65
Nγ		10.876
Fγs		1
Fγd		1
Fγi		0.717
B-term	kPa	152.028
Ultimate Bearing Capacity	kPa	1221.92
Factor of Safety for Bearing capacity		2
Allowable Bearing Capacity	kPa	610.96
Maximum Stress	kPa	175.133
Minimum Stress	kPa	162.908

Forces on Retaining Wall

Description	Fx	Fz	x coordinate	z coordinate
Weight of Wall		293.264	1.602	
Front Face Resistance	150.083	26.464	0	0.58
Force from Wedge		289.538	2.538	
Active Pressure	76.587	3.995	3.65	1.139
Weight of Soil on Front Face		30.154	0.256	

Stability Analysis Report

Parameter	Requirement	Actual Value	Result
Overturning, Factor of Safety	> 2	14.07	Pass
Sliding, Factor of Safety	> 2	4.848	Pass
Bearing Capacity, Eccentricity	< 0.608	0.022	Pass
Bearing Capacity, Max-Stress	< 610.9601	175.133	Pass

(b) Tabular Output

Figure: E8.5.5: Graphical and tabular Solution from the developed software

ANNEX-D: - Codes Used for the Development of the Software

Active Passive Calculations form

```
Imports System.Math
Public Class ActivePassiveCalculations

    Private Sub ActivePassiveCalculations_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim i, j As Integer

        Try
            WallPressureParametersView.DataSource = WallPressureParametersTable
            WallPressureParametersView.Columns(0).Width = 230
            WallPressureParametersView.Columns(1).Width = 110
            WallPressureParametersView.DefaultCellStyle.WrapMode =
DataGridViewTriState.True
            WallPressureParametersView.Columns(2).Width = 70


            ActiveSoilDisplayTable = ActiveSoilsTable.Copy
            For i = ActiveSoilDisplayTable.Rows.Count - 1 To 0 Step -1
                Dim SoilTypeNotAssigned As Boolean = True
                For j = 0 To AssignedSoilType.Rows.Count - 1 Step 1
                    If ActiveSoilDisplayTable.Rows(i).Item(0) =
ActiveSoilsTable.Rows(AssignedSoilType.Rows(j).Item(0)).Item(0) Then
                        SoilTypeNotAssigned = False
                    End If
                Next
                If SoilTypeNotAssigned Then ActiveSoilDisplayTable.Rows.RemoveAt(i)
            Next

            PassiveSoilDisplayTable = PassiveSoilTable.Copy
            If PassiveAssigned Then
                For i = PassiveSoilDisplayTable.Rows.Count - 1 To 0 Step -1
                    If PassiveSoilDisplayTable.Rows(i).Item(0) <>
PassiveSoilTable.Rows(PassiveSoilType).Item(0) Then
PassiveSoilDisplayTable.Rows.RemoveAt(i)
                Next
            Else
                PassiveSoilDisplayTable.Rows.Clear()
            End If


            For Each DataRow In ActiveSoilDisplayTable.Rows
                For i = 0 To ActiveSoilDisplayTable.Columns.Count - 1
                    Try
                        DataRow.item(i) = Round(DataRow.item(i), 3)
                    Catch ex As Exception
                    End Try
                Next
            Next DataRow

            For Each DataRow In PassiveSoilDisplayTable.Rows
                For i = 0 To PassiveSoilDisplayTable.Columns.Count - 1
                    Try
                        DataRow.item(i) = Round(DataRow.item(i), 3)
                    Catch ex As Exception
```

```
                End Try
            Next
        Next DataRow


        ActiveSoilsView.DataSource = ActiveSoilDisplayTable
        PassiveSoilsView.DataSource = PassiveSoilDisplayTable

        ActivePressureView.DataSource = ActivePressureTable
        PassivePressureView.DataSource = PassivePressureTable
    Catch ex As Exception
    End Try

End Sub

End Class
```

Calculations Table Form

```
Public Class CalculationsTable

    Private Sub CalculationsTable_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        CalculationsGridView.DataSource = CalculationsDataTable
        SoilsCalculationView.DataSource = EpSoilsTable
        EarthPressureParametersView.DataSource = CalculationParametersTable

        EarthPressureParametersView.Columns(0).Width = 250
        EarthPressureParametersView.Columns(1).Width = 80
        'EarthPressureParametersView.Columns(1).
        EarthPressureParametersView.DefaultCellStyle.WrapMode = DataGridViewTriState.True
        EarthPressureParametersView.Columns(2).Width = 70
    End Sub

End Class
```

Interface Input

```
Public Class InterfaceInput

    Private Sub ApplyButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ApplyButton.Click
        Try
            NewInterfaceDepth = TextBox1.Text
            DialogResult = DialogResult.OK
        Catch ex As Exception
        End Try
    End Sub

    Private Sub CancelInputButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CancelInputButton.Click
        DialogResult = DialogResult.Cancel
    End Sub

    Private Sub InterfaceInput_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        TextBox1.Select()
    End Sub

End Class
```

Main Form

```
Imports System.Math

Public Class MainForm

    Dim AssignmentTableCreated As Boolean = False
    Dim AssignmentTable As New DataTable
    'Dim NumberOfInterfaces As UInteger
    Dim GivenInterfaceTable, CalculationsInterfaceTable As New DataTable
    'Dim CalculateSoilsPropertiesTable As New DataTable
    Dim AnalysisCompleted As Boolean = False
    Dim GroundWaterDepth As Single = 0

    Dim UserEditBeta As Boolean = True
    Dim UserEditAlpha As Boolean = True
    Dim UserEditGWT As Boolean = True
    Dim UserEditSurcharge As Boolean = True

    Dim NumberOfAssignedRows As UInteger

    Dim DefaultCell As DataGridViewCell = New DataGridViewTextBoxCell()
    Dim LayerThicknessColumn As New DataGridViewColumn(DefaultCell)
    Dim LayerExtentsColumn As New DataGridViewColumn(DefaultCell)

    Dim SoilTypeColumn As New DataGridViewComboBoxColumn
    'Dim StabilityDescriptionColumn As New DataColumn("Description")

    Dim ProfileDrawingWidth, ProfileDrawingHeight As UInteger
    Dim SoilTypeMatrix() As Single

    'Dim FrictionAnglesMatrix As Single(,)
    Dim Surcharge As Single

    Dim WallType As String = "Gravity"
    Dim HeightsDataTable, WidthsDataTable As New DataTable

    Dim ForceFromWedge, WedgeLeverArm As Single
    Dim CheckDepth As Single
    Dim RetainingWallCalculated As Boolean = False
    Dim FLDistance, PadOverhang As Single
    Dim FrontFacePassiveAnalysis As Boolean = False
    Dim InitialRetainingDimensions As Boolean = True

    Dim HorizontalLever As Single
    Dim StabilityAnalysis As Boolean = False
    Dim WallArea, WallWeight, WallMoment, WallLeverArm As Single
    Dim ReactionLocation, eccentricity As Single
    Dim VerticalReaction, AntiSlidingForce As Single
    Dim UserWallDimensioning As Boolean = False
    Dim GraphicLever As Single
    Dim EpOutputTable, RetOutputTable As New DataTable
    Dim EarthPressureCalculationError As Boolean = False
    Dim count As Integer
    Dim TotalStressExists As Boolean
    Dim WallAngle As Single
```

```
    Dim TheoryNotSelected As Boolean = True
    Dim HydrostaticDueToCrackEnabled As Boolean = True
    Dim GwtFrontDiagramEnabled As Boolean = False

    Private Sub AddButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles AddButton.Click
        Dim i As Integer
        InterfaceInput.Text = "Add Interface"
        InterfaceInput.TextBox1.Text = ""
        My.Forms.InterfaceInput.ShowDialog()
        If InterfaceInput.DialogResult = DialogResult.OK Then
            For i = 0 To GivenInterfaceTable.Rows.Count - 1
                If GivenInterfaceTable.Rows(i).Item(0) = NewInterfaceDepth Then
                    InterfaceDataGridVeiw.Rows(i).Cells(0).Selected = True
                    Exit Sub
                End If
            Next
            If NewInterfaceDepth < 0 Then
                MsgBox("Value cannot be negative.")
                Exit Sub
            ElseIf NewInterfaceDepth > HeightOfAnalysis Then
                MsgBox("Value cannot be greater than Height.")
                Exit Sub
            End If

            GivenInterfaceTable.Rows.Add()

            'inserting the new interface depth at the proper location of the sequence
            i = GivenInterfaceTable.Rows.Count - 1
            Do Until GivenInterfaceTable.Rows(i - 1).Item(0) < NewInterfaceDepth
                GivenInterfaceTable.Rows(i).Item(0) = GivenInterfaceTable.Rows(i -
1).Item(0)
                i = i - 1
            Loop
            GivenInterfaceTable.Rows(i).Item(0) = NewInterfaceDepth
            InterfaceDataGridVeiw.ClearSelection()
            InterfaceDataGridVeiw.Rows(i).Cells(0).Selected = True

            CreateNewAssignSoilsGridView()
        End If

        EraseResults()
        Refresh()
    End Sub

    Private Sub MainForm_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

        'Arranging the interfaces DataGridView
        Try
            SoilDataInput.StressStateComboBox.SelectedIndex = 1
            SoilDataInput.WaterWeightComboBox.SelectedIndex = 0
            SoilDataInput.SoilTypeAtRestComboBox.SelectedIndex = 0
            AtRestRadioButton.Checked = True

            InterfaceDataGridVeiw.DataSource = GivenInterfaceTable
            GivenInterfaceTable.Columns.Add("Depth (from Top)", GetType(Single))
```

```
            GivenInterfaceTable.Rows.Add()
            GivenInterfaceTable.Rows.Add()
            GivenInterfaceTable.Rows(0).Item(0) = 0

            CalculationsInterfaceTable.Columns.Add("Depth", GetType(Single))

            InterfaceDataGridVeiw.ClearSelection()
            InterfaceDataGridVeiw.Columns(0).SortMode =
DataGridViewColumnSortMode.NotSortable
            HeightEntryBox.Text = 5
        Catch ex As Exception
        End Try

        'Arranging geometric properties
        AlphaTextBox.Text = 0
        BetaTextBox.Text = 0

        'Arranging the soils DataGridView
        Try
            SoilsDataGridView.DataSource = SoilDisplayTable
            SoilDisplayTable.Columns.Add("Type of Soil", GetType(String))
            SoilsDataGridView.Columns(0).SortMode =
DataGridViewColumnSortMode.NotSortable

            EpSoilsTable.Columns.Add("Type of Soil", GetType(String))         'column0
            EpSoilsTable.Columns.Add("Bulk Unit Weight", GetType(Single))     'column1
            EpSoilsTable.Columns.Add("Saturated Unit Weight", GetType(Single))
'column2
            EpSoilsTable.Columns.Add("Stress State", GetType(String))
'column3
            EpSoilsTable.Columns.Add("Angle of Backfill Soil", GetType(Single))
'column4
            EpSoilsTable.Columns.Add("Angle of Internal Friction", GetType(Single))
'column5
            EpSoilsTable.Columns.Add("Cohesion", GetType(Single))
'column6
            EpSoilsTable.Columns.Add("Angle of Soil Structure Friction", GetType(Single))
'column7
            EpSoilsTable.Columns.Add("Adhesion", GetType(Single)) 'column8
            EpSoilsTable.Columns.Add("Soil AtRest Condition", GetType(String))
'column9
            EpSoilsTable.Columns.Add("Poissons Ratio", GetType(Single))
'column10
            EpSoilsTable.Columns.Add("PI", GetType(Single))               'column11
            EpSoilsTable.Columns.Add("OCR", GetType(Single))          'column12
            EpSoilsTable.Columns.Add("Theory", GetType(String))   'column13
            EpSoilsTable.Columns.Add("Ko", GetType(Single))   'column14
            EpSoilsTable.Columns.Add("Ka", GetType(Single))   'column15
            EpSoilsTable.Columns.Add("Kp", GetType(Single))   'column16

        Catch ex As Exception
        End Try

        'Arranging the assignment DataGridView
        Try
            LayerThicknessColumn.ReadOnly = True
            LayerExtentsColumn.ReadOnly = True
            LayerThicknessColumn.SortMode = DataGridViewColumnSortMode.NotSortable
```

```
            LayerThicknessColumn.Width = 100
            LayerExtentsColumn.Width = 100
            SoilTypeColumn.Width = 200
            AssignDataGridView.Columns.Add(LayerThicknessColumn)
            AssignDataGridView.Columns.Add(LayerExtentsColumn)
            AssignDataGridView.Columns.Add(SoilTypeColumn)

            AssignDataGridView.Columns(0).Name = "Thickness"
            AssignDataGridView.Columns(1).Name = "Location"
            AssignDataGridView.Columns(2).Name = "Type of Soil"
            CreateNewAssignSoilsGridView()
        Catch ex As Exception
        End Try

        'Arranging the AssignedSoilType DataTable
        AssignedSoilType.Columns.Add("Soil Type Index", GetType(Integer))

        'Arranging Retaining Wall data input

        HeightsDataTable.Columns.Add()
        HeightsDataTable.Rows.Add()
        HeightsDataTable.Rows.Add()
        HeightsDataTable.Rows.Add()
        HeightsGridView.DataSource = HeightsDataTable
        WidthsDataTable.Columns.Add()
        WidthsDataTable.Rows.Add()
        WidthsDataTable.Rows.Add()
        WidthsDataTable.Rows.Add()
        WidthsGridView.DataSource = WidthsDataTable


        ResultForcesTable.Columns.Add("Description", GetType(String))
        ResultForcesTable.Columns.Add("Fx", GetType(Single))
        ResultForcesTable.Columns.Add("Fz", GetType(Single))
        ResultForcesTable.Columns.Add("x coordinate", GetType(Single))
        ResultForcesTable.Columns.Add("z coordinate", GetType(Single))


        For i = 0 To 4
            ResultForcesTable.Rows.Add()
        Next
        ResultForcesTable.Rows(0).Item(0) = "Weight of Wall"
        ResultForcesTable.Rows(1).Item(0) = "Front Face Resistance"
        ResultForcesTable.Rows(2).Item(0) = "Force from Wedge"
        ResultForcesTable.Rows(3).Item(0) = "Active Pressure"
        ResultForcesTable.Rows(4).Item(0) = "Weight of Soil on Front Face"

        SafetyFactorTable.Columns.Add("Parameter", GetType(String))
        SafetyFactorTable.Columns.Add("Requirement", GetType(String))
        SafetyFactorTable.Columns.Add("Actual Value", GetType(Single))
        SafetyFactorTable.Columns.Add("Result", GetType(String))
        For i = 1 To 4
            SafetyFactorTable.Rows.Add()
        Next
        SafetyFactorTable.Rows(0).Item(0) = "Overturning, Factor of Safety"
        SafetyFactorTable.Rows(1).Item(0) = "Sliding, Factor of Safety"
        SafetyFactorTable.Rows(2).Item(0) = "Bearing Capacity, Eccentricity"
        SafetyFactorTable.Rows(3).Item(0) = "Bearing Capacity, Max-Stress"
```

```
        'Arranging the StablilityAnalysis DataTable
        CalculationParametersTable.Columns.Add("Descritption", GetType(String))
        CalculationParametersTable.Columns.Add("Unit", GetType(String))
        CalculationParametersTable.Columns.Add("Magnitude", GetType(Single))

        'Arranging the StablilityAnalysis DataTable
        StabilityParametersTable.Columns.Add("Descritption", GetType(String))
        StabilityParametersTable.Columns.Add("Unit", GetType(String))
        StabilityParametersTable.Columns.Add("Magnitude", GetType(Single))

        'By defualt the gravity type of retaining wall is selected
        RetainingWallParameters.BCknownRadioButton.Checked = True
        GravityWallSelected()

        Refresh()
    End Sub

    Private Sub AddSoilsButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles AddSoilsButton.Click
        SoilNumber = EpSoilsTable.Rows.Count
        My.Forms.SoilDataInput.ShowDialog()
        If SoilDataInput.DialogResult = DialogResult.OK Then
            SoilDisplayTable.Rows.Add()
            SoilDisplayTable.Rows(SoilDisplayTable.Rows.Count - 1).Item(0) =
NewSoilLayerName
            SoilsDataGridView.ClearSelection()

            SoilTypeColumn.Items.Add(NewSoilLayerName)
            RetainingWallParameters.PassiveSoilTypeComboBox.Items.Add(NewSoilLayerName)
        End If

    End Sub

    Private Sub MainForm_Paint(ByVal sender As Object, ByVal e As
System.Windows.Forms.PaintEventArgs) Handles Me.Paint
        Dim SurfaceLoadPen As New Pen(Color.Red)

        Try
            If RetainingWallDiagram = False Then
                'Earth pressure analysis diagrams
                Dim SoilColor As New SolidBrush(Color.BurlyWood)
                Dim i As UInteger

                Dim ProfileOriginX, ProfileOriginY As UInteger
                Dim ProfileCoordinates As Point()
                Dim TopRightCoordinate, BottomLeftCoordinate As Point
                Dim HeightOfIncline, WallInclineDistance As UInteger

                If BackfillInclinationBeta = 0 Then
                    HeightOfIncline = 0
                Else
                    If WallInclinationAlpha = 0 Then
                        HeightOfIncline = 0.75 * 160 / (575 - 210) * (Me.Size.Height -
210) * _
                            BackfillInclinationBeta / (80 / 180 * Math.PI)
                        'at the 80deg backfill angle, the incline height will 75% of the
profile height
```

```
                    Else
                        Dim HeightAboveZero As Single
                        HeightAboveZero = FLDistance * Tan(BackfillInclinationBeta)
                        HeightOfIncline = 1.25 / (1.25 + HeightOfAnalysis /
HeightAboveZero) _
                            * 160 / (575 - 210) * (Me.Size.Height - 210)
                        'the full profile height(=160) = 1.25xHeightAbove0 +
HeightOfAnalysis, everything scalable
                    End If
                End If

                ProfileDrawingWidth = 200 / 800 * Me.Size.Width
                If WallInclinationAlpha = 0 Then
                    WallInclineDistance = 0
                Else
                    WallInclineDistance = 0.5 * ProfileDrawingWidth * _
                       WallInclinationAlpha / (80 / 180 * Math.PI)
                End If

                ProfileOriginX = 50 / 800 * Me.Size.Width
                ProfileOriginY = (110) / (575 - 210) * (Me.Size.Height - 210) +
HeightOfIncline
                ProfileDrawingHeight = (160) / (575 - 210) * (Me.Size.Height - 210) -
HeightOfIncline

                TopRightCoordinate = New Point(ProfileDrawingWidth, -HeightOfIncline)
                BottomLeftCoordinate = New Point(WallInclineDistance,
ProfileDrawingHeight)

                ProfileCoordinates = {New Point(0, 0), TopRightCoordinate, _
                                    New Point(ProfileDrawingWidth, ProfileDrawingHeight),
_
                                      BottomLeftCoordinate}

                'Drawing the soil profile graphic
                Dim SoilProfileGraphics As Graphics
                SoilProfileGraphics = Me.CreateGraphics
                Dim InterfacePen As New Pen(Color.Black, 2)
                Dim GroundWaterPen As New Pen(Color.Blue, 2)
                Dim WallPen As New Pen(Color.Brown)
                SoilProfileGraphics.TranslateTransform(ProfileOriginX, ProfileOriginY,
Drawing2D.MatrixOrder.Append)
                SoilProfileGraphics.FillPolygon(SoilColor, ProfileCoordinates)
                SoilProfileGraphics.DrawLine(WallPen, 0, 0, WallInclineDistance,
ProfileDrawingHeight)

                'Drawing the interface lines, GWT and respective elevations on the soil
profile
                Dim InterfaceY, InterfaceLeft As UInteger
                SoilProfileGraphics.DrawLine(InterfacePen, 0, 0, ProfileDrawingWidth, -
HeightOfIncline)
                SoilProfileGraphics.DrawString("0.00", font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=-40, y:=InterfaceY - 5)
                For i = 1 To GivenInterfaceTable.Rows.Count - 2
                    InterfaceY = GivenInterfaceTable.Rows(i).Item(0) / HeightOfAnalysis *
ProfileDrawingHeight
                    InterfaceLeft = WallInclineDistance *
GivenInterfaceTable.Rows(i).Item(0) / HeightOfAnalysis
```

```
                    SoilProfileGraphics.DrawLine(InterfacePen, InterfaceLeft, InterfaceY,
ProfileDrawingWidth, InterfaceY)

SoilProfileGraphics.DrawString(Format(GivenInterfaceTable.Rows(i).Item(0), "0.00"),
font:=SystemFonts.DefaultFont, brush:=Brushes.Black, x:=-40, y:=InterfaceY - 5)
                Next
                SoilProfileGraphics.DrawString(Format(GivenInterfaceTable.Rows(i).Item(0),
"0.00"), font:=SystemFonts.DefaultFont, brush:=Brushes.Black, x:=-40,
y:=ProfileDrawingHeight - 5)

                SoilProfileGraphics.DrawString(Format(GroundWaterDepth, "0.00"),
font:=SystemFonts.DefaultFont, brush:=Brushes.Black, x:=-40, y:=GroundWaterDepth /
HeightOfAnalysis * ProfileDrawingHeight - 5)
                SoilProfileGraphics.DrawLine(GroundWaterPen, WallInclineDistance *
GroundWaterDepth / HeightOfAnalysis _
                                             , GroundWaterDepth / HeightOfAnalysis *
ProfileDrawingHeight, ProfileDrawingWidth, GroundWaterDepth / HeightOfAnalysis *
ProfileDrawingHeight)
                SoilProfileGraphics.DrawString("G.W.T", font:=SystemFonts.DefaultFont,
brush:=Brushes.DarkBlue, x:=ProfileDrawingWidth + 5, y:=GroundWaterDepth /
HeightOfAnalysis * ProfileDrawingHeight - 5)

                FrontDiagramLabel.Text = "Soil Profile"
                FrontDiagramLabel.Left = 125 / 800 * Me.Size.Width
                FrontDiagramLabel.Top = 50 / (575 - 210) * (Me.Size.Height - 210)

                If AnalysisCompleted Then
                    Dim MaxStress As Single
                    'Dim MinStress As Single

                    For Each CalculationRow As DataRow In CalculationsDataTable.Rows
                        If CalculationRow.Item(22) > MaxStress Then MaxStress =
CalculationRow.Item(22)
                        If CalculationRow.Item(32) > MaxStress Then MaxStress =
CalculationRow.Item(32)
                    Next CalculationRow


                    Dim LateralPressureDiagram As Graphics
                    LateralPressureDiagram = Me.CreateGraphics
                    Dim LateralPressurePen As New Pen(Color.Red, 2)
                    Dim AxisPen As New Pen(Color.Black, 1)

                    'OutputGridView.Left = 580 / 800 * Me.Size.Width
                    OutputGridView.Top = 110 / (575 - 210) * (Me.Size.Height - 210)
                    OutputGridView.Height = 160 / (575 - 210) * (Me.Size.Height - 210)

                    Dim SpaceForGraphics As UInteger
                    Dim GapSace As UInteger
                    GapSace = 50 / 800 * Me.Size.Width
                    SpaceForGraphics = 0.5 * (OutputGridView.Location.X - (ProfileOriginX
+ ProfileDrawingWidth) - (3 * GapSace))

                    'Drawing the Horizontal Pressure Distribution
                    Dim Xscale, Yscale As Single
                    Dim NumberOfOutputRows As Integer
                    NumberOfOutputRows = EarthPressureTable.Rows.Count
                    If MaxStress <> 0 Then Xscale = SpaceForGraphics / MaxStress
```

```
                    Yscale = ProfileDrawingHeight /
CalculationsDataTable.Rows(NumberOfOutputRows - 1).Item(0)
                    LateralPressureDiagram.TranslateTransform(ProfileOriginX +
ProfileDrawingWidth + GapSace, _
                                                              ProfileOriginY,
Drawing2D.MatrixOrder.Append)

                    LateralPressureDiagram.DrawLine(AxisPen, 0,
CalculationsDataTable.Rows(0).Item(0) * Yscale, _
                                                     0, CInt(HeightOfAnalysis * Yscale))
                    LateralPressureDiagram.DrawLine(LateralPressurePen, 0,
CalculationsDataTable.Rows(0).Item(0) * Yscale _
                        , CalculationsDataTable.Rows(0).Item(22) * Xscale,
CalculationsDataTable.Rows(0).Item(0) * Yscale)
                    For i = 0 To NumberOfOutputRows - 2
                        LateralPressureDiagram.DrawLine(LateralPressurePen,
CalculationsDataTable.Rows(i).Item(22) * Xscale, _
                        CalculationsDataTable.Rows(i).Item(0) * Yscale,
CalculationsDataTable.Rows(i + 1).Item(22) * Xscale, _
                        CalculationsDataTable.Rows(i + 1).Item(0) * Yscale)
                    Next
                    LateralPressureDiagram.DrawLine(LateralPressurePen, 0,
HeightOfAnalysis * Yscale, _
                    CalculationsDataTable.Rows(NumberOfOutputRows - 1).Item(22) * Xscale,
HeightOfAnalysis * Yscale)

                    LateralPressureDiagram.DrawString("P-Horizontal= " &
Round(HorizontalResultantForce, 3) & " kN", font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=-10, y:=CInt(HeightOfAnalysis * Yscale) + 13)
                    If HorizontalResultantForce <> 0 Then _
                    LateralPressureDiagram.DrawString("Depth= " & Round(HeightOfAnalysis
- HorizontalResultantLocation, 3) _
                        & " m", font:=SystemFonts.DefaultFont, brush:=Brushes.Black, x:=-
10, y:=CInt(HeightOfAnalysis * Yscale) + 28)


                    'Drawing the Vertical Pressure Distribution
                    Dim VerticalPressureDiagram As Graphics
                    VerticalPressureDiagram = Me.CreateGraphics
                    VerticalPressureDiagram.TranslateTransform(ProfileOriginX +
ProfileDrawingWidth + GapSace + SpaceForGraphics + GapSace, _
                                                ProfileOriginY,
Drawing2D.MatrixOrder.Append)

                    VerticalPressureDiagram.DrawLine(AxisPen, 0,
CalculationsDataTable.Rows(0).Item(0) * Yscale, _
                                                     0, CInt(HeightOfAnalysis * Yscale))

                    If AnalysisType = "At Rest Earth Pressure" Then
                        'for at rest earth pressure, only the resultant force is drawn on
vertical pressure graph
                        Dim VerticalShift As Single
                        If WallInclinationAlpha <> 0 Then _
                            VerticalShift = WallInclineDistance / HeightOfAnalysis /
Tan(WallInclinationAlpha) * WedgeLeverArm
                        If ForceFromWedge <> 0 Then
                            VerticalPressureDiagram.DrawLine(LateralPressurePen,
CInt(VerticalShift), CInt(0.25 * HeightOfAnalysis * Yscale), _
```

```
                                    CInt(VerticalShift), CInt(0.75 * HeightOfAnalysis *
Yscale))
                            VerticalPressureDiagram.DrawLine(LateralPressurePen,
CInt(VerticalShift) - 5, CInt(0.75 * HeightOfAnalysis * Yscale) - 10, _
                                    CInt(VerticalShift), CInt(0.75 * HeightOfAnalysis *
Yscale))

                            VerticalPressureDiagram.DrawLine(LateralPressurePen,
CInt(VerticalShift) + 5, CInt(0.75 * HeightOfAnalysis * Yscale) - 10, _
                                                            CInt(VerticalShift),
CInt(0.75 * HeightOfAnalysis * Yscale))
                        End If
                        VerticalPressureDiagram.DrawString("P-Vertical= " &
Round(ForceFromWedge, 3) & " kN", font:=SystemFonts.DefaultFont, brush:=Brushes.Black,
x:=10, y:=HeightOfAnalysis * Yscale + 13)
                        If ForceFromWedge <> 0 Then _
                        VerticalPressureDiagram.DrawString("Shift= " &
Round(WedgeLeverArm, 3) & " m", font:=SystemFonts.DefaultFont, brush:=Brushes.Black,
x:=10, y:=HeightOfAnalysis * Yscale + 28)
                    Else
                        VerticalPressureDiagram.DrawLine(LateralPressurePen, 0,
CalculationsDataTable.Rows(0).Item(0) * Yscale _
                                                          ,
CalculationsDataTable.Rows(0).Item(32) * Xscale, CalculationsDataTable.Rows(0).Item(0) *
Yscale)
                        For i = 0 To NumberOfOutputRows - 2
                            VerticalPressureDiagram.DrawLine(LateralPressurePen,
CalculationsDataTable.Rows(i).Item(32) * Xscale, _
                            CalculationsDataTable.Rows(i).Item(0) * Yscale,
CalculationsDataTable.Rows(i + 1).Item(32) * Xscale, _
                            CalculationsDataTable.Rows(i + 1).Item(0) * Yscale)
                        Next
                        VerticalPressureDiagram.DrawLine(LateralPressurePen, 0,
HeightOfAnalysis * Yscale, _
                            CalculationsDataTable.Rows(NumberOfOutputRows - 1).Item(32) *
Xscale, HeightOfAnalysis * Yscale)

                        VerticalPressureDiagram.DrawString("P-Vertical= " &
Round(VerticalResultantForce, 3) & " kN", font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=10, y:=HeightOfAnalysis * Yscale + 13)
                        If VerticalResultantForce <> 0 Then _
                        VerticalPressureDiagram.DrawString("Shift= " &
Round(VerticalResultantLocation, 3) & " m", font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=10, y:=HeightOfAnalysis * Yscale + 28)
                    End If

                    'Preparing the numeric output table
                    OutputGridView.DataSource = EpOutputTable
                    OutputGridView.ClearSelection()
                    OutputGridView.Visible = True
                End If

                'Drawing the surcharge loads
                'Dim SurfaceLoadPen As New Pen(Color.Red)
                Dim x, y As Single
                If Surcharge <> 0 Then
                    SoilProfileGraphics.DrawLine(SurfaceLoadPen, 0, -20,
ProfileDrawingWidth, -HeightOfIncline - 20)
```

```
                    For i = 0 To 7
                        x = i / 7 * ProfileDrawingWidth
                        y = -i / 7 * HeightOfIncline
                        SoilProfileGraphics.DrawLine(SurfaceLoadPen, x, y, x, y - 20)
                        SoilProfileGraphics.DrawLine(SurfaceLoadPen, x, y, x - 5, y - 10)
                        SoilProfileGraphics.DrawLine(SurfaceLoadPen, x, y, x + 5, y - 10)
                    Next
                    SoilProfileGraphics.DrawString(Surcharge & " kPa",
font:=SystemFonts.DefaultFont, _
                                                     brush:=Brushes.Red, x:=x, y:=y - 35)
                End If
            Else
                'Retainingwall diagram enabled
                Dim SoilColor As New SolidBrush(Color.BurlyWood)
                Dim RetWallColor As New SolidBrush(Color.White)
                Dim WidthForGraphics, HeightForGraphics As Single
                Dim WidthScale, HeightScale, DiagramScale As Single
                Dim ProblemHeight, ProblemWidth As Single
                Dim GapSpace As Single = 25
                Dim DiagramMargin As Single
                Dim WallPen As New Pen(Color.Black, 2)
                Dim InterfacePen As New Pen(Color.Black, 1)
                Dim GroundWaterPen As New Pen(Color.Blue, 1)
                Dim DimensionPen As New Pen(Color.Green, 1)

                Dim NormalFormat As New StringFormat(ContentAlignment.MiddleCenter)
                Dim Left As New StringFormat(ContentAlignment.MiddleLeft)
                Dim Right As New StringFormat(ContentAlignment.MiddleRight)

                Dim TxtFont As New Font(FontFamily.GenericSansSerif, emSize:=7)
                NormalFormat.Alignment = StringAlignment.Center
                'NormalFormat.LineAlignment =StringAlignment.Center

                'TxtFont.Size = 10

                'VerticalPressureDiagram.DrawString("P-Vertical= " &
VerticalResultantForce & " kN", font:=SystemFonts.DefaultFont, brush:=Brushes.Black,
x:=10, y:=HeightOfAnalysis * Yscale + 13)

                FrontDiagramLabel.Text = "Retaining Wall"
                FrontDiagramLabel.Left = 125 / 800 * Me.Size.Width
                FrontDiagramLabel.Top = 50 / (575 - 210) * (Me.Size.Height - 210)

                Dim BasicWallGraphics, WallDimGraphics, WallForcesGraphics As Graphics
                BasicWallGraphics = Me.CreateGraphics
                WallDimGraphics = Me.CreateGraphics
                WallForcesGraphics = Me.CreateGraphics


                OutputGridView.Top = 100 / (575 - 210) * (Me.Size.Height - 210)
                OutputGridView.Left = Me.Size.Width - 220
                WidthForGraphics = (OutputGridView.Location.X - 4 * GapSpace) / 3.0
                HeightForGraphics = 175 / (575 - 210) * (Me.Size.Height - 210)
                OutputGridView.Height = HeightForGraphics

                Dim InclineHeight As Integer
                If BackfillInclinationBeta = 0 Then
                    InclineHeight = 0
```

```
                Else
                    InclineHeight = H / 5 / (575 - 210) * (Me.Size.Height - 210)
                End If

                If GwtFrontDiagramEnabled And (GroundWaterDepthFront -
PassiveSoilThickness) <= b Then
                    ProblemHeight = Max(H + b / 5 + InclineHeight, H +
(GroundWaterDepthFront - PassiveSoilThickness) + InclineHeight)
                Else
                    ProblemHeight = H + b / 5 + InclineHeight
                End If
                ProblemWidth = b / 5 + b + b / 5

                HeightScale = HeightForGraphics / ProblemHeight
                WidthScale = WidthForGraphics / ProblemWidth

                If HeightScale < WidthScale Then
                    DiagramScale = HeightScale
                    DiagramMargin = 90 / (575 - 210) * (Me.Size.Height - 210)
                Else
                    DiagramScale = WidthScale
                    DiagramMargin = 90 / (575 - 210) * (Me.Size.Height - 210) _
                        + 0.75 * (HeightForGraphics - DiagramScale * ProblemHeight)
                End If

                Dim WallCoordinates As Point()
                Dim ActiveSidePolygon As Point()
                Dim x1, x2, x3, x4, x5, x6, y1, y2, y3, y4 As Single
                x1 = -0.5 * b
                x2 = x1 + b1
                x3 = x1 + b1 + S1 * (H - d)
                x4 = x1 + b1 + S1 * (H - d) + b2
                x5 = x1 + b1 + S1 * (H - d) + b2 + S2 * (H - d)
                x6 = 0.5 * b
                y1 = InclineHeight
                y2 = InclineHeight + (H - d)
                y3 = InclineHeight + H
                If GwtFrontDiagramEnabled And (GroundWaterDepthFront -
PassiveSoilThickness) <= b Then
                    y4 = Max(InclineHeight + H + b / 5, InclineHeight + H +
(GroundWaterDepthFront - PassiveSoilThickness))
                Else
                    y4 = InclineHeight + H + b / 5
                End If


                WallCoordinates = {New Point(DiagramScale * x3, DiagramScale * y1), _
                                   New Point(DiagramScale * x2, DiagramScale * y2), _
                                   New Point(DiagramScale * x1, DiagramScale * y2), _
                                   New Point(DiagramScale * x1, DiagramScale * y3), _
                                   New Point(DiagramScale * x6, DiagramScale * y3), _
                                   New Point(DiagramScale * x6, DiagramScale * y2), _
                                   New Point(DiagramScale * x5, DiagramScale * y2), _
                                   New Point(DiagramScale * x4, DiagramScale * y1)}

                ActiveSidePolygon = {New Point(DiagramScale * x4, DiagramScale * y1), _
                 New Point(DiagramScale * x4, DiagramScale * y4), _
                New Point(DiagramScale * (x6 + b / 5), DiagramScale * y4), _
```

```
                New Point(DiagramScale * (x6 + b / 5), 0)}


                BasicWallGraphics.TranslateTransform(GapSpace + 0.5 * WidthForGraphics, _
                          DiagramMargin, Drawing2D.MatrixOrder.Append)


                BasicWallGraphics.FillRectangle(SoilColor, DiagramScale * (x1 - b / 5),
DiagramScale * (InclineHeight + H - PassiveSoilThickness), _
                                                DiagramScale * (b / 5 + x3 - x1),
DiagramScale * (PassiveSoilThickness + (y4 - y3)))

                BasicWallGraphics.FillRectangle(SoilColor, DiagramScale * (x1 - b / 5),
DiagramScale * (InclineHeight + H), _
                                DiagramScale * (2 * b / 5 + b), DiagramScale * (y4 - y3))

                BasicWallGraphics.FillPolygon(SoilColor, ActiveSidePolygon)


                'Drawing the interface lines and GWT line
                BasicWallGraphics.DrawLine(InterfacePen, DiagramScale * x4, DiagramScale
* InclineHeight, _
                                           DiagramScale * (b / 2 + b / 5), 0)
                BasicWallGraphics.DrawLine(InterfacePen, DiagramScale * (x1 - b / 5),
DiagramScale * (H - PassiveSoilThickness + InclineHeight), _
                           DiagramScale * x3, DiagramScale * (H - PassiveSoilThickness +
InclineHeight))
                'BasicWallGraphics.DrawString("+" & Format(PassiveSoilThickness, "0.00"),
font:=SystemFonts.DefaultFont, brush:=Brushes.Black, x:=DiagramScale * (x1 - b / 5) - 37,
y:=DiagramScale * (y3 - PassiveSoilThickness) - 5)

                'BasicWallGraphics.DrawString("+" & Format(H, "0.00"),
font:=SystemFonts.DefaultFont, brush:=Brushes.Black, x:=DiagramScale * (x6 + b / 5) + 2,
y:=DiagramScale * y1 - 11)
                For i = 1 To GivenInterfaceTable.Rows.Count - 2 Step 1
                    BasicWallGraphics.DrawLine(InterfacePen, DiagramScale * x4,
DiagramScale * (y1 + GivenInterfaceTable.Rows(i).Item(0)), _
                       DiagramScale * (b / 2 + b / 5), DiagramScale * (y1 +
GivenInterfaceTable.Rows(i).Item(0)))
                    'BasicWallGraphics.DrawString("+" & Format(H -
GivenInterfaceTable.Rows(i).Item(0), "0.00"), font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=DiagramScale * (x6 + b / 5) + 2, y:=DiagramScale * (y1 +
GivenInterfaceTable.Rows(i).Item(0)) - 11)
                Next

                BasicWallGraphics.DrawLine(GroundWaterPen, DiagramScale * x4,
DiagramScale * (y1 + GroundWaterDepth), _
                   DiagramScale * (b / 2 + b / 5), DiagramScale * (y1 + GroundWaterDepth))
                BasicWallGraphics.DrawString("G.W.T", font:=DefaultFont,
brush:=Brushes.Blue, x:=DiagramScale * (x6 + b / 5) + 2, y:=DiagramScale * (y1 +
GroundWaterDepth) - 10)

                If GwtFrontDiagramEnabled And (GroundWaterDepthFront -
PassiveSoilThickness) <= b Then
                    BasicWallGraphics.DrawLine(GroundWaterPen, DiagramScale * x3,
DiagramScale * (y3 - PassiveSoilThickness + GroundWaterDepthFront), _
                                       DiagramScale * (x1 - b / 5), DiagramScale * (y3 -
PassiveSoilThickness + GroundWaterDepthFront))
```

```
                    BasicWallGraphics.DrawString("G.W.T", font:=DefaultFont, 
brush:=Brushes.Blue, x:=DiagramScale * (x1 - b / 5) - 40, y:=DiagramScale * (y3 - 
PassiveSoilThickness + GroundWaterDepthFront) - 10)
                End If

                Dim x, y As Single
                'Drawing the Active side surface loads
                If Surcharge <> 0 Then
                    BasicWallGraphics.DrawLine(SurfaceLoadPen, DiagramScale * x4, 
DiagramScale * y1 - 15, DiagramScale * (x6 + b / 5), -15)
                    For i = 0 To 7
                        x = DiagramScale * (x4 + i / 7 * (x6 - x4 + b / 5))
                        y = (y1 - i / 7 * InclineHeight) * DiagramScale
                        BasicWallGraphics.DrawLine(SurfaceLoadPen, x, y, x, y - 15)
                        BasicWallGraphics.DrawLine(SurfaceLoadPen, x, y, x - 3, y - 8)
                        BasicWallGraphics.DrawLine(SurfaceLoadPen, x, y, x + 3, y - 8)
                    Next
                    BasicWallGraphics.DrawString(Surcharge & " kPa", 
font:=SystemFonts.DefaultFont, _
                                                brush:=Brushes.Red, x:=x + 5, y:=y - 20)
                End If

                'Drawing the Passive side surface loads
                If PassiveSurcharge <> 0 Then
                    BasicWallGraphics.DrawLine(SurfaceLoadPen, DiagramScale * (x1 - b / 
5), _
                                                DiagramScale * (y3 - PassiveSoilThickness) 
- 15, _
                                                DiagramScale * x3, _
                                                DiagramScale * (y3 - PassiveSoilThickness) 
- 15)
                    For i = 0 To 4
                        x = DiagramScale * ((x1 - b / 5) + (x3 - x1 + b / 5) * i / 4)
                        y = DiagramScale * (y3 - PassiveSoilThickness)
                        BasicWallGraphics.DrawLine(SurfaceLoadPen, x, y, x, y - 15)
                        BasicWallGraphics.DrawLine(SurfaceLoadPen, x, y, x - 3, y - 8)
                        BasicWallGraphics.DrawLine(SurfaceLoadPen, x, y, x + 3, y - 8)
                    Next
                    BasicWallGraphics.DrawString(PassiveSurcharge & " kPa", 
font:=SystemFonts.DefaultFont, _
                                                brush:=Brushes.Red, x:=DiagramScale * (x1 
- b / 5) - 20, y:=y - 30)
                End If

                BasicWallGraphics.FillPolygon(RetWallColor, WallCoordinates)
                BasicWallGraphics.DrawPolygon(WallPen, WallCoordinates)

                'Drawing the coordinate axes
                Dim OriginPen As New Pen(Color.Magenta, 2)
                Dim OriginY As Integer
                Dim AxisFont As New Font(FontFamily.GenericSansSerif, emSize:=11)


                BasicWallGraphics.DrawLine(OriginPen, DiagramScale * x1 - 7, DiagramScale 
* y3 + 1, _
                                            DiagramScale * x1 + 30, DiagramScale * y3 + 1)
                BasicWallGraphics.DrawLine(OriginPen, DiagramScale * x1 + 20, 
DiagramScale * y3 - 5, _
```

```
                                         DiagramScale * x1 + 30, DiagramScale * y3)
                BasicWallGraphics.DrawLine(OriginPen, DiagramScale * x1 + 20,
DiagramScale * y3 + 5, _
                                         DiagramScale * x1 + 30, DiagramScale * y3)
                BasicWallGraphics.DrawString("+x", font:=AxisFont, brush:=Brushes.Magenta,
x:=DiagramScale * x1 + 26, y:=DiagramScale * y3 + 1)

                OriginY = Max(30, DiagramScale * 1.4 * d)
                BasicWallGraphics.DrawLine(OriginPen, DiagramScale * x1, DiagramScale *
y3 + 7, _
                       DiagramScale * x1, DiagramScale * y3 - OriginY)
                BasicWallGraphics.DrawLine(OriginPen, DiagramScale * x1 + 5, DiagramScale
* y3 - OriginY + 7, _
                       DiagramScale * x1, DiagramScale * y3 - OriginY)
                BasicWallGraphics.DrawLine(OriginPen, DiagramScale * x1 - 5, DiagramScale
* y3 - OriginY + 7, _
                       DiagramScale * x1, DiagramScale * y3 - OriginY)
                BasicWallGraphics.DrawString("+z", font:=AxisFont, brush:=Brushes.Magenta,
x:=DiagramScale * x1 - 11, y:=DiagramScale * y3 - OriginY - 18)


                'Drawing the diagram with basic dimensions
                WallDimGraphics.TranslateTransform(GapSpace + WidthForGraphics + 1.4 *
GapSpace + 0.5 * WidthForGraphics, _
                          DiagramMargin, Drawing2D.MatrixOrder.Append)
                WallDimGraphics.DrawPolygon(WallPen, WallCoordinates)


                'Top dimension
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x3, DiagramScale *
y1 - 15, DiagramScale * x4, DiagramScale * y1 - 15)

                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x3, DiagramScale *
y1 - 5, DiagramScale * x3, DiagramScale * y1 - 20)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x3 + 3,
DiagramScale * y1 - 18, DiagramScale * x3 - 3, DiagramScale * y1 - 12)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x4, DiagramScale *
y1 - 5, DiagramScale * x4, DiagramScale * y1 - 20)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x4 + 3,
DiagramScale * y1 - 18, DiagramScale * x4 - 3, DiagramScale * y1 - 12)

                WallDimGraphics.DrawString(b2, font:=TxtFont, brush:=Brushes.Black,
x:=(DiagramScale * 0.5 * (x3 + x4)), y:=DiagramScale * y1 - 35, format:=NormalFormat)


                'Dimension on left side (heights and depths)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 15,
DiagramScale * y1, DiagramScale * x1 - 15, DiagramScale * y3)

                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 20,
DiagramScale * y1, DiagramScale * x1 - 5, DiagramScale * y1)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 18,
DiagramScale * y1 + 3, DiagramScale * x1 - 12, DiagramScale * y1 - 3)

                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 20,
DiagramScale * y2, DiagramScale * x1 - 5, DiagramScale * y2)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 18,
DiagramScale * y2 + 3, DiagramScale * x1 - 12, DiagramScale * y2 - 3)
```

```
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 20,
DiagramScale * y3, DiagramScale * x1 - 5, DiagramScale * y3)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 18,
DiagramScale * y3 + 3, DiagramScale * x1 - 12, DiagramScale * y3 - 3)

                WallDimGraphics.FillRectangle(RetWallColor, x:=CInt(DiagramScale * x1 -
16), y:=CInt(0.5 * DiagramScale * (y1 + y2)) - 1, width:=5, height:=13)
                WallDimGraphics.DrawString(Format(H - d, "0.000"), font:=TxtFont,
brush:=Brushes.Black, x:=(DiagramScale * x1 - 15), y:=0.5 * DiagramScale * (y1 + y2),
format:=NormalFormat)
                WallDimGraphics.FillRectangle(RetWallColor, x:=CInt(DiagramScale * x1 -
16), y:=CInt(0.5 * DiagramScale * (y2 + y3)) - 4, width:=5, height:=13)
                WallDimGraphics.DrawString(Format(d, "0.000"), font:=TxtFont,
brush:=Brushes.Black, x:=(DiagramScale * x1 - 15), y:=0.5 * DiagramScale * (y2 + y3) - 3,
format:=NormalFormat)

                'Full height dimension on the left side
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 15 - 25,
DiagramScale * y1, DiagramScale * x1 - 15 - 25, DiagramScale * y3)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 40 - 5,
DiagramScale * y1, DiagramScale * x1 - 40 + 5, DiagramScale * y1)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 40 - 3,
DiagramScale * y1 + 3, DiagramScale * x1 - 40 + 3, DiagramScale * y1 - 3)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 40 - 5,
DiagramScale * y3, DiagramScale * x1 - 40 + 5, DiagramScale * y3)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 40 - 3,
DiagramScale * y3 + 3, DiagramScale * x1 - 40 + 3, DiagramScale * y3 - 3)
                WallDimGraphics.FillRectangle(RetWallColor, x:=CInt(DiagramScale * x1) -
40, y:=CInt(0.5 * DiagramScale * (y1 + y3)) + 9, width:=5, height:=13)
                WallDimGraphics.DrawString(Format(H, "0.00"), font:=TxtFont,
brush:=Brushes.Black, x:=(DiagramScale * x1 - 40), y:=0.5 * DiagramScale * (y1 + y3) + 10,
format:=NormalFormat)

                'Bottom dimensions
                WallDimGraphics.TranslateTransform(0, DiagramScale * (H + InclineHeight),
Drawing2D.MatrixOrder.Append)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1, 15,
DiagramScale * x6, 15)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1, 5, DiagramScale
* x1, 20)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 2, 18,
DiagramScale * x1 + 3, 13)

                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x2, 5, DiagramScale
* x2, 20)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x2 - 2, 18,
DiagramScale * x2 + 3, 13)

                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x5, 5, DiagramScale
* x5, 20)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x5 - 2, 18,
DiagramScale * x5 + 3, 13)

                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x6, 5, DiagramScale
* x6, 20)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x6 - 2, 18,
DiagramScale * x6 + 3, 13)
```

```
                WallDimGraphics.DrawString(Round(b1, 3), font:=TxtFont,
brush:=Brushes.Black, x:=(DiagramScale * 0.5 * (x1 + x2)) - 5, y:=20,
format:=NormalFormat)
                WallDimGraphics.DrawString(Round(b - b1 - b3, 3), font:=TxtFont,
brush:=Brushes.Black, x:=(DiagramScale * 0.5 * (x4 + x2)) + 3, y:=27,
format:=NormalFormat)
                WallDimGraphics.DrawString(Round(b3, 3), font:=TxtFont,
brush:=Brushes.Black, x:=(DiagramScale * 0.5 * (x5 + x6)), y:=20, format:=Left)

                '*******


                'Bottom baseline dimension
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1, 15 + 35,
DiagramScale * x6, 15 + 35)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1, 5 + 35,
DiagramScale * x1, 20 + 35)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x1 - 2, 18 + 35,
DiagramScale * x1 + 3, 13 + 35)

                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x6, 5 + 35,
DiagramScale * x6, 20 + 35)
                WallDimGraphics.DrawLine(DimensionPen, DiagramScale * x6 - 2, 18 + 35,
DiagramScale * x6 + 3, 13 + 35)
                WallDimGraphics.DrawString(Round(b, 3), font:=TxtFont,
brush:=Brushes.Black, x:=(DiagramScale * 0.5 * (x1 + x6)), y:=15 + 35,
format:=NormalFormat)


                ''Drawing the Retaining Wall with Forces Diagram
                Dim ForcesPen As New Pen(Color.Red, 2)
                Dim ReactionPen As New Pen(Color.Magenta, 2)
                Dim WedgePen As New Pen(Color.Gray, 1)

                If RetainingWallCalculated Then
                    WallForcesGraphics.TranslateTransform(GapSpace + WidthForGraphics +
GapSpace + WidthForGraphics + GapSpace + 0.5 * WidthForGraphics, _
        DiagramMargin, Drawing2D.MatrixOrder.Append)

                    'Drawing the passive soil area
                    If ResultForcesTable.Rows(4).Item(2) <> 0 Then
                        WallForcesGraphics.DrawLine(WedgePen, DiagramScale * x1,
DiagramScale * y3, DiagramScale * x1, DiagramScale * (y3 - PassiveSoilThickness))
                        WallForcesGraphics.DrawLine(WedgePen, DiagramScale * x1,
DiagramScale * (y3 - PassiveSoilThickness), DiagramScale * x3, DiagramScale * (y3 -
PassiveSoilThickness))
                    End If

                    WallForcesGraphics.FillPolygon(RetWallColor, WallCoordinates)
                    WallForcesGraphics.DrawPolygon(WallPen, WallCoordinates)

                    WallForcesGraphics.TranslateTransform(-0.5 * DiagramScale * b,
DiagramScale * y3, _
    Drawing2D.MatrixOrder.Append)


                    'Drawing the wedge & force due to weight (if nonzero)
```

```
                    If ResultForcesTable.Rows(2).Item(2) <> 0 Then
                        'drawing the wedge area
                        Dim y0 As Integer
                        y0 = InclineHeight - CInt(InclineHeight / (b / 5 + x6 - x4) * (x6
- x4))
                        WallForcesGraphics.DrawLine(WedgePen, DiagramScale * b, -
DiagramScale * d, DiagramScale * b, -DiagramScale * (H + InclineHeight))
                        WallForcesGraphics.DrawLine(WedgePen, DiagramScale * (x4 - x1), -
DiagramScale * H, DiagramScale * b, -DiagramScale * (H + InclineHeight))


                        'drawing the wt. of wedge arrow
                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
ResultForcesTable.Rows(2).Item(3), -DiagramScale * CInt(0.5 * H - d), _
                              DiagramScale * ResultForcesTable.Rows(2).Item(3), -
DiagramScale * CInt(0.5 * H - d) - 30)

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
ResultForcesTable.Rows(2).Item(3), _
                        -DiagramScale * CInt(0.5 * H - d), DiagramScale *
ResultForcesTable.Rows(2).Item(3) + 4, _
                        -DiagramScale * CInt(0.5 * H - d) - 8)

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
ResultForcesTable.Rows(2).Item(3), _
                        -DiagramScale * CInt(0.5 * H - d), DiagramScale *
ResultForcesTable.Rows(2).Item(3) - 4, _
                        -DiagramScale * CInt(0.5 * H - d) - 8)


WallForcesGraphics.DrawString(Round(ResultForcesTable.Rows(2).Item(2), 3) & " kN/m",
font:=TxtFont, brush:=Brushes.Black, _
    x:=DiagramScale * ResultForcesTable.Rows(2).Item(3) + 5, y:=-DiagramScale * CInt(0.5
* H - d) - 42, format:=NormalFormat)
                    End If

                    'Drawing the weight of wall
                    WallForcesGraphics.DrawLine(ForcesPen, DiagramScale * WallLeverArm, _
                                            DiagramScale * (-H / 2), _
                                            DiagramScale * (WallLeverArm), _
                                            DiagramScale * (-H / 2) - 40)
                    WallForcesGraphics.DrawLine(ForcesPen, DiagramScale * (WallLeverArm)
- 5, _
                                            DiagramScale * (-H / 2) - 7, _
                                            DiagramScale * (WallLeverArm), _
                                            DiagramScale * (-H / 2))
                    WallForcesGraphics.DrawLine(ForcesPen, DiagramScale * (WallLeverArm)
+ 5, _
                                            DiagramScale * (-H / 2) - 7, _
                                            DiagramScale * (WallLeverArm), _
                                            DiagramScale * (-H / 2))
                    WallForcesGraphics.DrawString(Round(WallWeight, 3) & " kN/m",
font:=TxtFont, brush:=Brushes.Black, _
    x:=DiagramScale * WallLeverArm, y:=DiagramScale * (-H / 2) - 50, format:=Right)

                    'Drawing the ground reaction
                    x = ReactionLocation
                    Dim Qtoe, Qheel, L, R As Single
```

```
                    Dim Begin As Single = 0
                    Dim Length As Single = DiagramScale * b
                    Qtoe = VerticalReaction / b * (1 + 6 * eccentricity / b)
                    Qheel = VerticalReaction / b * (1 - 6 * eccentricity / b)
                    If Qtoe > 0 Or Qheel > 0 Then
                        If Qtoe < 0 Then
                            Begin = DiagramScale * b * Qtoe / (Qtoe - Qheel)
                            L = 0
                            R = 30
                            WallForcesGraphics.DrawString(Round(Qheel, 3) & " kPa",
font:=TxtFont, brush:=Brushes.Black, _
                                                          x:=DiagramScale * b - 10, y:=30
+ 5)
                        ElseIf Qheel < 0 Then
                            Length = DiagramScale * b * Qtoe / (Qtoe - Qheel)
                            L = 30
                            R = 0
                            WallForcesGraphics.DrawString(Round(Qtoe, 3) & " kPa",
font:=TxtFont, brush:=Brushes.Black, _
                                                          x:=-10, y:=30 + 5)
                        Else
                            L = 30 * Qtoe / Max(Qtoe, Qheel)
                            R = 30 * Qheel / Max(Qtoe, Qheel)
                            WallForcesGraphics.DrawString(Round(Qheel, 3) & " kPa",
font:=TxtFont, brush:=Brushes.Black, _
                          x:=DiagramScale * b - 10, y:=R + 5)
                            WallForcesGraphics.DrawString(Round(Qtoe, 3) & " kPa",
font:=TxtFont, brush:=Brushes.Black, _
                          x:=-10, y:=L + 5, format:=Right)

                        End If
                        WallForcesGraphics.DrawLine(ReactionPen, Begin, L, Begin + Length,
R)
                        For i = 0 To 5
                            x = Begin + Length * i / 5.0
                            y = L + (R - L) / Length * x
                            WallForcesGraphics.DrawLine(ReactionPen, x, 0, x, y)
                            If y > 0 Then
                                WallForcesGraphics.DrawLine(ReactionPen, x, 0, x - 5, 5)
                                WallForcesGraphics.DrawLine(ReactionPen, x, 0, x + 5, 5)
                            End If
                        Next
                    End If

                    'Drawing the resultant for front face (passive) earth pressure
(horizontal & vertical)
                    If ResultForcesTable.Rows(1).Item(1) <> 0 Then
                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(1).Item(3)), _
                               -DiagramScale * (ResultForcesTable.Rows(1).Item(4)), _
                               DiagramScale * (ResultForcesTable.Rows(1).Item(3)) - 30,
DiagramScale * (-ResultForcesTable.Rows(1).Item(4)))

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(1).Item(3)), DiagramScale * (-ResultForcesTable.Rows(1).Item(4)),
_
    DiagramScale * (ResultForcesTable.Rows(1).Item(3)) - 8, DiagramScale * (-
ResultForcesTable.Rows(1).Item(4)) + 4)
```

```
                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(1).Item(3)),
    DiagramScale * (-ResultForcesTable.Rows(1).Item(4)), DiagramScale *
(ResultForcesTable.Rows(1).Item(3)) - 8, _
    DiagramScale * (-ResultForcesTable.Rows(1).Item(4)) - 4)


WallForcesGraphics.DrawString(Round(ResultForcesTable.Rows(1).Item(1), 3) & " kN/m",
font:=TxtFont, brush:=Brushes.Black, _
                                                   x:=DiagramScale *
(ResultForcesTable.Rows(1).Item(3)) - 30, y:=DiagramScale * (-
ResultForcesTable.Rows(1).Item(4)), format:=NormalFormat)
                    End If

                    If ResultForcesTable.Rows(1).Item(2) <> 0 And False Then
                        'intentionally skipped condition
                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(1).Item(3)), _
                                DiagramScale * (-ResultForcesTable.Rows(1).Item(4)) - 2,
_
                                DiagramScale * (ResultForcesTable.Rows(1).Item(3)),
DiagramScale * (-ResultForcesTable.Rows(1).Item(4)) - 27)

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(1).Item(3)), _
        DiagramScale * (-ResultForcesTable.Rows(1).Item(4)) - 2, _
        DiagramScale * (ResultForcesTable.Rows(1).Item(3)) - 4, DiagramScale * (-
ResultForcesTable.Rows(1).Item(4)) - 8)

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(1).Item(3)), _
        DiagramScale * (-ResultForcesTable.Rows(1).Item(4)) - 2, _
        DiagramScale * (ResultForcesTable.Rows(1).Item(3)) + 4, DiagramScale * (-
ResultForcesTable.Rows(1).Item(4)) - 8)


WallForcesGraphics.DrawString(Round(ResultForcesTable.Rows(1).Item(2), 3) & " kN/m",
font:=TxtFont, brush:=Brushes.Black, _
                           x:=DiagramScale * (ResultForcesTable.Rows(1).Item(3)) - 15,
y:=DiagramScale * (-ResultForcesTable.Rows(1).Item(4)) - 35, format:=NormalFormat)
                    End If



                    'Drawing the force due to weight of passive area soil
                    If ResultForcesTable.Rows(4).Item(2) <> 0 Then
                        'If ResultForcesTable.Rows(1).Item(2) <> 0 Then
                        y = -(d + 0.8 * (PassiveSoilThickness - d))
                        x = DiagramScale * ResultForcesTable.Rows(4).Item(3)

                        WallForcesGraphics.DrawLine(ForcesPen, x, _
                                DiagramScale * y, x, DiagramScale * y - 40)

                        WallForcesGraphics.DrawLine(ForcesPen, x, _
                                DiagramScale * y, x - 4, DiagramScale * y - 8)

                        WallForcesGraphics.DrawLine(ForcesPen, x, _
```

```
                                DiagramScale * y, x + 4, DiagramScale * y - 8)


WallForcesGraphics.DrawString(Round(ResultForcesTable.Rows(4).Item(2), 3) & " kN/m",
font:=TxtFont, brush:=Brushes.Black, _
                           x:=x - 15, y:=DiagramScale * y - 50, format:=NormalFormat)
                    End If

                    'Drawing the resultant for retained Earth Pressure (horizontal and
vertical)
                    'Horizontal Force
                    If ResultForcesTable.Rows(3).Item(1) <> 0 Then
                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(3).Item(3)), _
                                DiagramScale * (-GraphicLever), _
                               DiagramScale * (ResultForcesTable.Rows(3).Item(3)) + 30, _
                                 DiagramScale * (-GraphicLever))

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(3).Item(3)), _
        DiagramScale * (-GraphicLever), _
       DiagramScale * (ResultForcesTable.Rows(3).Item(3)) + 8, _
         DiagramScale * (-GraphicLever) + 4)

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(3).Item(3)), _
        DiagramScale * (-GraphicLever), _
       DiagramScale * (ResultForcesTable.Rows(3).Item(3)) + 8, _
         DiagramScale * (-GraphicLever) - 4)


WallForcesGraphics.DrawString(Round(ResultForcesTable.Rows(3).Item(1), 3) & " kN/m",
font:=TxtFont, brush:=Brushes.Black, _
    x:=DiagramScale * (ResultForcesTable.Rows(3).Item(3)) + 30, y:=DiagramScale * (-
GraphicLever) + 5, format:=NormalFormat)
                    End If

                    'Vertical Force
                    If ResultForcesTable.Rows(3).Item(2) <> 0 Then
                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(3).Item(3)), _
                                DiagramScale * (-GraphicLever) - 2, _
                               DiagramScale * (ResultForcesTable.Rows(3).Item(3)), _
                                 DiagramScale * (-GraphicLever) - 30)

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(3).Item(3)), _
        DiagramScale * (-GraphicLever) - 2, _
       DiagramScale * (ResultForcesTable.Rows(3).Item(3)) - 4, _
         DiagramScale * (-GraphicLever) - 8)

                        WallForcesGraphics.DrawLine(ForcesPen, DiagramScale *
(ResultForcesTable.Rows(3).Item(3)), _
        DiagramScale * (-GraphicLever) - 2, _
       DiagramScale * (ResultForcesTable.Rows(3).Item(3)) + 4, _
         DiagramScale * (-GraphicLever) - 8)
```

```
WallForcesGraphics.DrawString(Round(ResultForcesTable.Rows(3).Item(2), 3) & " kN/m",
font:=TxtFont, brush:=Brushes.Black, _
    x:=DiagramScale * (ResultForcesTable.Rows(3).Item(3)) + 1, y:=DiagramScale * (-
GraphicLever) - 30, format:=Right)


                        OutputGridView.DataSource = RetOutputTable
                        OutputGridView.ClearSelection()
                        OutputGridView.Visible = True
                    End If
                End If
            End If

        Catch ex As Exception
        End Try

    End Sub

    Private Sub RunAnalysisButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RunAnalysisButton.Click

        Dim i As Integer
        For i = 0 To AssignDataGridView.Rows.Count - 1
            If AssignDataGridView.Rows(i).Cells(2).Value = "" Then
                MsgBox("Assignment of Soil Types for the profile is incomplete.")
                Exit Sub
            End If
        Next

        CalculateSoilsPropertiesTable = EpSoilsTable.Copy
        CalculationsInterfaceTable = GivenInterfaceTable.Copy

        If Theory = "Coulomb" Then
            FLDistance = 0
        Else
            FLDistance = HeightOfAnalysis * Tan(WallInclinationAlpha)
        End If


        CalculationsDataTable.Columns.Clear()
        CalculationsDataTable.Rows.Clear()

        EarthPressureCalculationError = False
        CalculateEarthPressure()
        If EarthPressureCalculationError Then Exit Sub

        EarthPressureTable = CalculationsDataTable.Copy

        EpOutputTable.Columns.Clear()
        EpOutputTable.Rows.Clear()
        EpOutputTable.Columns.Add("Depth", GetType(Single))
        EpOutputTable.Columns.Add("Horizontal Component", GetType(Single))
        For i = 0 To EarthPressureTable.Rows.Count - 1
            EpOutputTable.Rows.Add()
            EpOutputTable.Rows(i).Item(0) = Round(EarthPressureTable.Rows(i).Item(0), 3)
            EpOutputTable.Rows(i).Item(1) = Round(EarthPressureTable.Rows(i).Item(22), 3)
        Next
```

```
        If AnalysisType <> "At Rest Earth Pressure" Then
            EpOutputTable.Columns.Add("Vertical Component", GetType(Single))
            For i = 0 To EarthPressureTable.Rows.Count - 1
                EpOutputTable.Rows(i).Item(2) = Round(EarthPressureTable.Rows(i).Item(32),
3)
            Next
        End If

        AnalysisCompleted = True
        Refresh()
        ResultForcesTable.Rows(2).Item(2) = ForceFromWedge

        'Rounding to 3 decimal places
        For Each DataRow In CalculationParametersTable.Rows
            Try
                DataRow.item(2) = Round(DataRow.item(2), 3)
            Catch ex As Exception
            End Try
        Next DataRow

        For Each DataRow In EpSoilsTable.Rows
            For i = 0 To EpSoilsTable.Columns.Count - 1
                Try
                    DataRow.item(i) = Round(DataRow.item(i), 3)
                Catch ex As Exception
                End Try
            Next
        Next DataRow

        For Each DataRow In CalculationsDataTable.Rows
            For i = 0 To CalculationsDataTable.Columns.Count - 1
                Try
                    DataRow.item(i) = Round(DataRow.item(i), 3)
                Catch ex As Exception
                End Try
            Next
        Next DataRow
    End Sub

    Private Sub HeightEntryBox_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles HeightEntryBox.TextChanged
        Try
            If HeightEntryBox.Text > 0 And HeightEntryBox.Text <> HeightOfAnalysis Then
                HeightOfAnalysis = HeightEntryBox.Text
                H = HeightOfAnalysis

                If PassiveSoilThickness > H Then PassiveSoilThickness = H

                GivenInterfaceTable.Rows(GivenInterfaceTable.Rows.Count - 1).Item(0) =
HeightOfAnalysis

                UserWallDimensioning = False
                SelectWallType()
                RefreshRetainingGridViews()
                UserWallDimensioning = True

                GivenInterfaceTable.Rows.Clear()
                GivenInterfaceTable.Rows.Add()
```

```
                GivenInterfaceTable.Rows.Add()
                GivenInterfaceTable.Rows(0).Item(0) = 0
                GivenInterfaceTable.Rows(1).Item(0) = HeightOfAnalysis
                CreateNewAssignSoilsGridView()
                GroundWaterDepth = 0

                EraseResults()
                Refresh()
            End If
        Catch ex As Exception
        End Try

        'GivenInterfaceTable.Rows.Add()
        'GivenInterfaceTable.Rows(1).Item(0) = HeightOfAnalysis
    End Sub

    Private Sub NewToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles NewToolStripMenuItem.Click
        Application.Restart()
    End Sub

    Private Sub GroundWaterTextBox_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles GroundWaterTextBox.TextChanged
        Try
            If GroundWaterTextBox.Text >= 0 Then
                'If GroundWaterTextBox.Text >= 0 And GroundWaterTextBox.Text <=
HeightOfAnalysis Then
                If GroundWaterTextBox.Text > HeightOfAnalysis Then
GroundWaterTextBox.Text = HeightOfAnalysis

                GroundWaterDepth = GroundWaterTextBox.Text
                If UserEditGWT Then EraseResults()
                Refresh()
            End If
        Catch ex As Exception
        End Try
    End Sub

    Private Sub CalculationTableToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles CalculationTableToolStripMenuItem.Click
        If RetainingWallDiagram Then
            My.Forms.ActivePassiveCalculations.Show()
        Else
            My.Forms.CalculationsTable.Show()
        End If
    End Sub

    Private Sub MainForm_Resize(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Resize
        Refresh()
    End Sub

    Private Sub GroundWater_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles GroundWater.Enter
        UserEditGWT = False
        GroundWaterTextBox.Text = GroundWaterDepth
        UserEditGWT = True
        GroundWaterTextBox.Select()
```

```
        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub AlphaTextBox_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles AlphaTextBox.TextChanged
        Try
            If AlphaTextBox.Text >= 0 And AlphaTextBox.Text < 90 Then
                WallInclinationAlpha = Math.PI / 180 * AlphaTextBox.Text
                WallAngle = WallInclinationAlpha
                FLDistance = HeightOfAnalysis * Tan(WallInclinationAlpha)
                If UserEditAlpha Then EraseResults()
                Refresh()
            End If
        Catch ex As Exception
        End Try
    End Sub

    Private Sub BetaTextBox_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BetaTextBox.TextChanged
        Try
            If BetaTextBox.Text >= 0 And BetaTextBox.Text < 90 Then
                BackfillInclinationBeta = Math.PI / 180 * BetaTextBox.Text
                If UserEditBeta Then EraseResults()
                Refresh()
            Else
            End If
        Catch ex As Exception
        End Try
    End Sub

    Private Sub GeometyTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles GeometyTab.Enter
        UserEditAlpha = False
        AlphaTextBox.Text = Round(WallInclinationAlpha * 180 / Math.PI, 4)
        UserEditAlpha = True
        AlphaTextBox.Select()

        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub GeometyTab_Paint(ByVal sender As Object, ByVal e As
System.Windows.Forms.PaintEventArgs) Handles GeometyTab.Paint
        Dim InclinedWallGraphics As Graphics
        Dim DrawingPen As New Pen(Color.Black)
        Dim GrayPen As New Pen(Color.Gray)
        InclinedWallGraphics = GeometyTab.CreateGraphics
        Dim SoilColor As New SolidBrush(Color.BurlyWood)

        InclinedWallGraphics.TranslateTransform(350, 40, Drawing2D.MatrixOrder.Append)
        'InclinedWallGraphics.DrawLine(DrawingPen, 0, 0, 75, 0)
        Dim PointA As New Point(-50, 100)
        Dim PointB As New Point(0, 0)
        Dim PointC As New Point(35, 0)
```

```
        Dim PointD As New Point(85, 100)
        Dim PointE As New Point(175, 100)
        Dim PointF As New Point(175, 0)

        InclinedWallGraphics.FillPolygon(SoilColor, {PointC, PointD, PointE, PointF})
        InclinedWallGraphics.DrawPolygon(DrawingPen, {PointA, PointB, PointC, PointD})
        InclinedWallGraphics.DrawLines(DrawingPen, {PointC, PointF})
        InclinedWallGraphics.DrawLines(DrawingPen, {PointD, PointE})

        InclinedWallGraphics.DrawLines(GrayPen, {PointC, New Point(35, 35)})
        InclinedWallGraphics.DrawLines(GrayPen, {PointD, New Point(85, 65)})
        InclinedWallGraphics.DrawLines(GrayPen, {New Point(35, 25), New Point(47, 22)})
        InclinedWallGraphics.DrawLines(GrayPen, {New Point(85, 75), New Point(73, 78)})
        InclinedWallGraphics.DrawString("α", font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=40, y:=35)
        InclinedWallGraphics.DrawString("α", font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=70, y:=50)

    End Sub

    Private Sub GroundTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles GroundTab.Enter
        UserEditBeta = False
        BetaTextBox.Text = Round(BackfillInclinationBeta * 180 / Math.PI, 4)
        UserEditBeta = True

        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        BetaTextBox.Select()
        Refresh()
    End Sub

    Private Sub GroundTab_Paint(ByVal sender As Object, ByVal e As
System.Windows.Forms.PaintEventArgs) Handles GroundTab.Paint
        Dim InclinedBackfillGraphics As Graphics
        Dim DrawingPen As New Pen(Color.Black)
        Dim GrayPen As New Pen(Color.Gray)
        InclinedBackfillGraphics = GroundTab.CreateGraphics
        Dim SoilColor As New SolidBrush(Color.BurlyWood)

        InclinedBackfillGraphics.TranslateTransform(350, 50, Drawing2D.MatrixOrder.Append)
        'InclinedWallGraphics.DrawLine(DrawingPen, 0, 0, 75, 0)
        Dim PointA As New Point(-5, 100)
        Dim PointB As New Point(0, 0)
        Dim PointC As New Point(50, 0)
        Dim PointD As New Point(55, 100)
        Dim PointE As New Point(190, 100)
        Dim PointF As New Point(190, -25)

        InclinedBackfillGraphics.FillPolygon(SoilColor, {PointC, PointD, PointE, PointF})
        InclinedBackfillGraphics.DrawPolygon(DrawingPen, {PointA, PointB, PointC, PointD})
        InclinedBackfillGraphics.DrawLines(DrawingPen, {PointC, PointF})
        InclinedBackfillGraphics.DrawLines(DrawingPen, {PointD, PointE})

        InclinedBackfillGraphics.DrawLines(GrayPen, {PointC, New Point(150, 0)})
        InclinedBackfillGraphics.DrawLines(GrayPen, {New Point(140, 0), New Point(140, -
6), _
```

```
                                                        New Point(139, -12), New Point(139,
-17)})
        InclinedBackfillGraphics.DrawString("ß", font:=SystemFonts.DefaultFont,
brush:=Brushes.Black, x:=160, y:=-15)
    End Sub

    Private Sub RankineButton_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RankineButton.CheckedChanged
        If RankineButton.Checked Then
            EraseResults()

            Theory = "Rankine"
            TheoryTypeLabel.Text = " - Rankine Earth Pressure Theory"

            BetaTextBox.Enabled = True
            AlphaTextBox.Enabled = True

            If EpSoilsTable.Rows.Count > 0 Then
                MsgBox("Soil data has to be entered and assigned like new.")
            End If
            EpSoilsTable.Rows.Clear()
            SoilDisplayTable.Rows.Clear()
            CreateNewAssignSoilsGridView()
            RetainingWallParameters.PassiveSoilTypeComboBox.Items.Clear()
            SoilTypeColumn.Items.Clear()
        End If
    End Sub

    Private Sub BellTheory_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BellTheory.CheckedChanged
        If BellTheory.Checked Then
            EraseResults()

            Theory = "Bell"
            TheoryTypeLabel.Text = " - Bell's Solution for Cohesive/mixed Soils"

            BetaTextBox.Text = 0
            BetaTextBox.Enabled = False
            AlphaTextBox.Enabled = True

            If EpSoilsTable.Rows.Count > 0 Then
                MsgBox("Soil data has to be entered and assigned like new.")
            End If
            EpSoilsTable.Rows.Clear()
            SoilDisplayTable.Rows.Clear()
            CreateNewAssignSoilsGridView()
            RetainingWallParameters.PassiveSoilTypeComboBox.Items.Clear()
            SoilTypeColumn.Items.Clear()
        Else
            BetaTextBox.Enabled = True
        End If
    End Sub

    Private Sub CoulombTheory_CheckedChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CoulombTheory.CheckedChanged
        If CoulombTheory.Checked Then
            EraseResults()
```

```
            Theory = "Coulomb"
            TheoryTypeLabel.Text = " - Coulomb's Earth Pressure Theory"


            If EpSoilsTable.Rows.Count > 0 Then
                MsgBox("Soil data has to be entered and assigned like new.")
            End If
            EpSoilsTable.Rows.Clear()
            SoilDisplayTable.Rows.Clear()
            CreateNewAssignSoilsGridView()
            RetainingWallParameters.PassiveSoilTypeComboBox.Items.Clear()
            SoilTypeColumn.Items.Clear()
        End If
    End Sub

    Private Sub AtRestRadioButton_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles AtRestRadioButton.CheckedChanged

        If AtRestRadioButton.Checked Then
            AnalysisType = "At Rest Earth Pressure"

            GroupBox3.Visible = False
            Theory = ""
            TheoryTypeLabel.Visible = False
            AnalysisTypeLabel.Text = "At Rest Earth Pressure"

            AlphaTextBox.Enabled = True
            BetaTextBox.Enabled = True
        Else
            TheoryTypeLabel.Visible = True
        End If

        EraseResults()

        If EpSoilsTable.Rows.Count > 0 Then
            MsgBox("Soil data has to be entered and assigned like new.")
        End If
        EpSoilsTable.Rows.Clear()
        SoilDisplayTable.Rows.Clear()
        CreateNewAssignSoilsGridView()
        RetainingWallParameters.PassiveSoilTypeComboBox.Items.Clear()
        SoilTypeColumn.Items.Clear()

        TensionCrackCheckBox.Visible = False

    End Sub

    Private Sub ActiveRadioButton_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ActiveRadioButton.CheckedChanged
        If ActiveRadioButton.Checked Then AnalysisType = "Active Earth Pressure"
        EraseResults()
        GroupBox3.Text = "Theory for Active Earth Pressure Calculation"
        GroupBox3.Visible = True
        AnalysisTypeLabel.Text = "Active Earth Pressure"

        If TheoryNotSelected Then RankineButton.Checked = True
        TheoryNotSelected = False
```

```
        Try
            EpSoilsTable.Columns(12).ColumnName = "Theory"
        Catch ex As Exception
        End Try

        TensionCrackCheckBox.Visible = True
    End Sub

    Private Sub PassiveRadioButton_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles PassiveRadioButton.CheckedChanged
        If PassiveRadioButton.Checked Then AnalysisType = "Passive Earth Pressure"
        EraseResults()

        GroupBox3.Text = "Theory for Passive Earth Pressure Calculation"
        GroupBox3.Visible = True
        AnalysisTypeLabel.Text = "Passive Earth Pressure"

        If TheoryNotSelected Then RankineButton.Checked = True
        TheoryNotSelected = False

        Try
            EpSoilsTable.Columns(12).ColumnName = "Theory"
        Catch ex As Exception
        End Try

        TensionCrackCheckBox.Visible = True
    End Sub

    Private Sub KeriselAbsiTheory_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles KeriselAbsiTheory.CheckedChanged
        If KeriselAbsiTheory.Checked Then
            EraseResults()

            Theory = "KeriselAbsi"
            TheoryTypeLabel.Text = "Kerisel and Absi Earth Pressure Theory"
            BetaTextBox.Text = 0
            BetaTextBox.Enabled = False
            AlphaTextBox.Text = 0
            AlphaTextBox.Enabled = False

            If EpSoilsTable.Rows.Count > 0 Then
                MsgBox("Soil data has to be entered and assigned like new.")
            End If
            EpSoilsTable.Rows.Clear()
            SoilDisplayTable.Rows.Clear()
            CreateNewAssignSoilsGridView()
            RetainingWallParameters.PassiveSoilTypeComboBox.Items.Clear()
            SoilTypeColumn.Items.Clear()
        Else
            BetaTextBox.Enabled = True
            AlphaTextBox.Enabled = True
        End If
    End Sub

    Private Sub SurchargeTextBox_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles SurchargeTextBox.TextChanged
        Try
            If SurchargeTextBox.Text >= 0 Then
```

```
                Surcharge = SurchargeTextBox.Text
                If UserEditSurcharge Then EraseResults()
                Refresh()
            End If
        Catch ex As Exception
        End Try
    End Sub

    Private Sub SurchargeTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles SurchargeTab.Enter
        UserEditSurcharge = False
        SurchargeTextBox.Text = Surcharge
        UserEditSurcharge = True
        SurchargeTextBox.Select()

        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub RetWallTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles RetWallTab.Enter

        ActiveSoilsTable = EpSoilsTable.Copy
        PassiveSoilTable = EpSoilsTable

        HeightBox.Text = HeightOfAnalysis

        If AnalysisType = "At Rest Earth Pressure" Then
            RetWallButton.Enabled = False
        Else
            RetWallButton.Enabled = True
        End If

        RefreshRetainingGridViews()
        UserWallDimensioning = True

        RetainingWallDiagram = True
        If RetainingWallCalculated = False Then
            OutputGridView.Visible = False
        Else
            OutputGridView.DataSource = RetOutputTable
            OutputGridView.Visible = True
        End If

        ArrangeSoilPropertiesForRetWall()

        HeightsGridView.ClearSelection()
        WidthsGridView.ClearSelection()
        Refresh()

    End Sub

    Private Sub RetWallTab_Paint(ByVal sender As Object, ByVal e As
System.Windows.Forms.PaintEventArgs) Handles RetWallTab.Paint

        '************
        Dim RetWallGraphics As Graphics
```

```
        Dim WallPen As New Pen(Color.Black)
        Dim DimensionPen As New Pen(Color.Green)

        If AnalysisType = "At Rest Earth Pressure" Then
            Label26.Visible = True
        Else
            Label26.Visible = False
        End If

        RetWallGraphics = RetWallTab.CreateGraphics
        RetWallGraphics.TranslateTransform(190, 40, Drawing2D.MatrixOrder.Append)

        Select Case WallType
            Case "Gravity"
                RetWallGraphics.DrawLines(WallPen, {New Point(35, 0), New Point(20, 80)})
                RetWallGraphics.DrawLines(WallPen, {New Point(20, 80), New Point(0, 80)})
                RetWallGraphics.DrawLines(WallPen, {New Point(0, 80), New Point(0, 100)})
                RetWallGraphics.DrawLines(WallPen, {New Point(0, 100), New Point(80,
100)})
                RetWallGraphics.DrawLines(WallPen, {New Point(80, 100), New Point(80,
80)})
                RetWallGraphics.DrawLines(WallPen, {New Point(80, 80), New Point(60, 80)})
                RetWallGraphics.DrawLines(WallPen, {New Point(60, 80), New Point(50, 0)})
                RetWallGraphics.DrawLines(WallPen, {New Point(50, 0), New Point(35, 0)})

                RetWallGraphics.DrawString("b1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=3, y:=65)
                RetWallGraphics.DrawString("b2", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=35, y:=-17)
                RetWallGraphics.DrawString("b3", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=63, y:=65)
                RetWallGraphics.DrawString("d", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=83, y:=82)

                RetWallGraphics.DrawLines(DimensionPen, {New Point(20, 53), New Point(20,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(20, 27), New Point(25,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(25, 27), New Point(20,
53)})
                RetWallGraphics.DrawString("S1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=15, y:=13)
                RetWallGraphics.DrawString("1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=10, y:=30)

                RetWallGraphics.DrawLines(DimensionPen, {New Point(62, 53), New Point(62,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(62, 27), New Point(57,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(57, 27), New Point(62,
53)})
                RetWallGraphics.DrawString("S2", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=53, y:=13)
                RetWallGraphics.DrawString("1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=62, y:=30)

            Case "Cantilever"
                RetWallGraphics.DrawLines(WallPen, {New Point(30, 0), New Point(20, 85)})
```

```
                RetWallGraphics.DrawLines(WallPen, {New Point(20, 85), New Point(0, 85)})
                RetWallGraphics.DrawLines(WallPen, {New Point(0, 85), New Point(0, 100)})
                RetWallGraphics.DrawLines(WallPen, {New Point(0, 100), New Point(80,
100)})
                RetWallGraphics.DrawLines(WallPen, {New Point(80, 100), New Point(80,
85)})
                RetWallGraphics.DrawLines(WallPen, {New Point(80, 85), New Point(43, 85)})
                RetWallGraphics.DrawLines(WallPen, {New Point(43, 85), New Point(40, 0)})
                RetWallGraphics.DrawLines(WallPen, {New Point(40, 0), New Point(30, 0)})

                RetWallGraphics.DrawString("b1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=3, y:=68)
                RetWallGraphics.DrawString("b2", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=30, y:=-17)
                RetWallGraphics.DrawString("b3", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=53, y:=68)
                RetWallGraphics.DrawString("d", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=83, y:=85)

                RetWallGraphics.DrawLines(DimensionPen, {New Point(17, 53), New Point(17,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(17, 27), New Point(22,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(22, 27), New Point(17,
53)})
                RetWallGraphics.DrawString("S1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=11, y:=13)
                RetWallGraphics.DrawString("1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=7, y:=30)

                RetWallGraphics.DrawLines(DimensionPen, {New Point(50, 53), New Point(50,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(50, 27), New Point(45,
27)})
                RetWallGraphics.DrawLines(DimensionPen, {New Point(45, 27), New Point(50,
53)})
                RetWallGraphics.DrawString("S2", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=41, y:=13)
                RetWallGraphics.DrawString("1", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=50, y:=30)
        End Select

        RetWallGraphics.DrawLines(DimensionPen, {New Point(-16, 0), New Point(-8, 0)})
        RetWallGraphics.DrawLines(DimensionPen, {New Point(-16, 100), New Point(-8, 100)})
        RetWallGraphics.DrawLines(DimensionPen, {New Point(-14, 2), New Point(-10, -2)})
        RetWallGraphics.DrawLines(DimensionPen, {New Point(-14, 102), New Point(-10, 98)})
        RetWallGraphics.DrawLines(DimensionPen, {New Point(-12, 0), New Point(-12, 40)})
        RetWallGraphics.DrawLines(DimensionPen, {New Point(-12, 60), New Point(-12, 100)})
        RetWallGraphics.DrawString("H", font:=SystemFonts.DefaultFont,
brush:=Brushes.Green, x:=-18, y:=45)
    End Sub

    Private Sub RetWallButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RetWallButton.Click
        My.Forms.RetainingWallParameters.ShowDialog()

        If RetainingWallParameters.DialogResult = DialogResult.OK Then
            PassiveAssigned = True
```

```
            CheckStabilityButton.Enabled = True
            GwtFrontDiagramEnabled = True

            EraseResults()
            Refresh()
        End If
    End Sub

    Private Sub CalculateEarthPressure()

        Dim i, j, k As Integer
        Dim RowNumber As Integer
        Dim ProgressiveTensileCrack As Boolean = True
        Dim WaterInfiltrationDepth As Single

        Try
            CalculationParametersTable.Rows.Clear()

            CalculationParametersTable.Rows.Add()
            RowNumber = CalculationParametersTable.Rows.Count - 1
            CalculationParametersTable.Rows(RowNumber).Item(0) = "Height of Analysis"
            CalculationParametersTable.Rows(RowNumber).Item(1) = "m"
            CalculationParametersTable.Rows(RowNumber).Item(2) = HeightOfAnalysis

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 1).Item(0) = "Ground Inclination"
            CalculationParametersTable.Rows(RowNumber + 1).Item(1) = "Deg"
            CalculationParametersTable.Rows(RowNumber + 1).Item(2) = Round(180 *
BackfillInclinationBeta / Math.PI, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 2).Item(0) = "Wall Inclination"
            CalculationParametersTable.Rows(RowNumber + 2).Item(1) = "Deg"
            CalculationParametersTable.Rows(RowNumber + 2).Item(2) = Round(180 *
WallInclinationAlpha / Math.PI, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 3).Item(0) = "Depth of Ground
Water"
            CalculationParametersTable.Rows(RowNumber + 3).Item(1) = "m"
            CalculationParametersTable.Rows(RowNumber + 3).Item(2) = GroundWaterDepth

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 4).Item(0) = "Surface Loads"
            CalculationParametersTable.Rows(RowNumber + 4).Item(1) = "kPa"
            CalculationParametersTable.Rows(RowNumber + 4).Item(2) = Surcharge

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 6).Item(0) = AnalysisType


            CalculationsDataTable.Columns.Add("z coordinate", GetType(Single))
'Column 0
            CalculationsDataTable.Columns.Add("Soil Type", GetType(String))
'Column 1
            CalculationsDataTable.Columns.Add("Bulk Unit Weight", GetType(Single))
'Column 2
```

```
            CalculationsDataTable.Columns.Add("Saturated Unit Weight", GetType(Single))
'Column 3
            CalculationsDataTable.Columns.Add("Stress State", GetType(String))
'column 4
            CalculationsDataTable.Columns.Add("Angle of Internal Friction",
GetType(String))    'column 5
            CalculationsDataTable.Columns.Add("Cohesion", GetType(Single))
'column 6
            CalculationsDataTable.Columns.Add("Angle of Soil-Structure Friction",
GetType(Single))  'Column 7
            CalculationsDataTable.Columns.Add("Adhesion", GetType(Single))
'Column 8
            CalculationsDataTable.Columns.Add("K-at rest", GetType(Single))
'Column 9
            CalculationsDataTable.Columns.Add("Component for Cohesive Soils",
GetType(Single))  'column 10

            Dim SoilNumber As UInteger
            j = 0
            For i = 0 To (CalculationsInterfaceTable.Rows.Count - 1) - 1

                CalculationsDataTable.Rows.Add()
                CalculationsDataTable.Rows(j).Item(0) =
CalculationsInterfaceTable.Rows(i).Item(0)

                If FrontFacePassiveAnalysis Then
                    Try
                        CalculationsDataTable.Rows(j).Item(1) =
CalculateSoilsPropertiesTable.Rows(PassiveSoilType).Item("Type of Soil")
                        SoilNumber =
RetainingWallParameters.PassiveSoilTypeComboBox.SelectedIndex
                    Catch ex As Exception
                        MsgBox("Soil Type has not been assigned to front face of
retaining wall!")
                        EarthPressureCalculationError = True
                        Exit Sub
                    End Try
                Else
                    Try
                        CalculationsDataTable.Rows(j).Item(1) =
AssignDataGridView.Rows(i).Cells(2).Value
                        SoilNumber = AssignedSoilType.Rows(i).Item(0)
                    Catch ex As Exception
                        MsgBox("Assignment of soil types for the profile is incomplete!")
                        EarthPressureCalculationError = True
                        Exit Sub
                    End Try
                End If

                CalculationsDataTable.Rows(j).Item(2) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("Bulk Unit Weight")
                CalculationsDataTable.Rows(j).Item(3) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("Saturated Unit Weight")
                CalculationsDataTable.Rows(j).Item(4) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("Stress State")
                CalculationsDataTable.Rows(j).Item(5) = Math.PI / 180 *
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("Angle of Internal Friction")
```

```
                CalculationsDataTable.Rows(j).Item(6) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("Cohesion")
                Try
                    CalculationsDataTable.Rows(j).Item(7) = Math.PI / 180 *
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("Angle of Soil Structure Friction")
                Catch ex As Exception
                End Try
                Try
                    CalculationsDataTable.Rows(j).Item(8) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("Adhesion")
                Catch ex As Exception
                End Try

                CalculationsDataTable.Rows(j).Item(10) = 0

                Select Case AnalysisType
                    Case "Active Earth Pressure"
                        CalculationsDataTable.Columns(9).ColumnName = "K-active"
                        CalculationsDataTable.Rows(j).Item(9) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("ka")
                        If CalculationsDataTable.Rows(j).Item(6) <> 0 Then
                            If Theory = "Bell" Then CalculationsDataTable.Rows(j).Item(10)
= -2 * CalculationsDataTable.Rows(j).Item(6) * Sqrt(CalculationsDataTable.Rows(j).Item(9))
                            If Theory = "Coulomb" Or Theory = "KeriselAbsi" Then
CalculationsDataTable.Rows(j).Item(10) = -2 * CalculationsDataTable.Rows(j).Item(6) *
Sqrt(CalculationsDataTable.Rows(j).Item(9) * (1 + CalculationsDataTable.Rows(j).Item(8) /
CalculationsDataTable.Rows(j).Item(6)))

                        End If
                    Case "Passive Earth Pressure"
                        CalculationsDataTable.Columns(9).ColumnName = "K-passive"
                        CalculationsDataTable.Rows(j).Item(9) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("kp")
                        If CalculationsDataTable.Rows(j).Item(6) <> 0 Then
                            If Theory = "Bell" Then CalculationsDataTable.Rows(j).Item(10)
= 2 * CalculationsDataTable.Rows(j).Item(6) * Sqrt(CalculationsDataTable.Rows(j).Item(9))
                            If Theory = "Coulomb" Or Theory = "KeriselAbsi" Then
CalculationsDataTable.Rows(j).Item(10) = 2 * CalculationsDataTable.Rows(j).Item(6) *
Sqrt(CalculationsDataTable.Rows(j).Item(9) * (1 + CalculationsDataTable.Rows(j).Item(8) /
CalculationsDataTable.Rows(j).Item(6)))

                        End If
                    Case "At Rest Earth Pressure"
                        CalculationsDataTable.Columns(9).ColumnName = "K-at rest"
                        CalculationsDataTable.Rows(j).Item(9) =
CalculateSoilsPropertiesTable.Rows(SoilNumber).Item("ko")
                End Select
                j = j + 1

                'Copying values for bottom end of interface
                CalculationsDataTable.Rows.Add()
                CalculationsDataTable.Rows(j).Item(0) = CalculationsInterfaceTable.Rows(i
+ 1).Item(0)
                For k = 1 To CalculationsDataTable.Columns.Count - 1
                    CalculationsDataTable.Rows(j).Item(k) = CalculationsDataTable.Rows(j
- 1).Item(k)
                Next
                j = j + 1
```

```
            Next

            'Inserting a row above z=0 to consider the top of fictitious line
            CalculationsDataTable.Rows.Add()
            For i = CalculationsDataTable.Rows.Count - 1 To 1 Step -1
                For k = 0 To CalculationsDataTable.Columns.Count - 1
                    CalculationsDataTable.Rows(i).Item(k) = CalculationsDataTable.Rows(i
- 1).Item(k)
                Next
            Next

            CalculationsDataTable.Rows(0).Item(0) = CalculationsDataTable.Rows(1).Item(0)
_
            - FLDistance * Tan(BackfillInclinationBeta)


            'Inserting a row at the depth of ground water (if necessary)
            If GroundWaterDepth <
CalculationsDataTable.Rows(CalculationsDataTable.Rows.Count - 1).Item(0) Then
                j = 0
                Do Until CalculationsDataTable.Rows(j).Item(0) >= GroundWaterDepth
                    j = j + 1
                Loop
                If CalculationsDataTable.Rows(j).Item(0) <> GroundWaterDepth Then
                    CalculationsDataTable.Rows.Add()
                    For i = CalculationsDataTable.Rows.Count - 2 To j Step -1
                        For k = 0 To CalculationsDataTable.Columns.Count - 1
                            CalculationsDataTable.Rows(i + 1).Item(k) =
CalculationsDataTable.Rows(i).Item(k)
                        Next
                    Next
                    CalculationsDataTable.Rows(j).Item(0) = GroundWaterDepth
                End If
            End If

            CalculationsDataTable.Columns.Add("Depth from GWT", GetType(Single))
'Column 11
            CalculationsDataTable.Columns.Add("Effective Unit Weight", GetType(Single))
'Column 12

            For Each CalculationRow As DataRow In CalculationsDataTable.Rows()
                If GroundWaterDepth >= CalculationRow.Item(0) Then
                    'soil above ground water table
                    CalculationRow.Item(11) = 0

                    If CalculationRow.Item(4) = "Effective Stress" Then
                        'effective stress case
                        CalculationRow.Item(12) = CalculationRow.Item(2)
                    Else
                        'Total stress case
                        CalculationRow.Item(12) = CalculationRow.Item(3)
                    End If

                Else
                    'soil below ground water table
                    CalculationRow.Item(11) = CalculationRow.Item(0) - GroundWaterDepth
                    If CalculationRow.Item(4) = "Effective Stress" Then
                        'effective stress case
```

```
                        CalculationRow.Item(12) = CalculationRow.Item(3) -
UnitWeightOfWater
                    Else
                        'Total stress case
                        CalculationRow.Item(12) = CalculationRow.Item(3)
                    End If
                End If
            Next CalculationRow

            CalculationsDataTable.Columns.Add("Segment Thickness", GetType(Single))
'Column 13
            CalculationsDataTable.Columns.Add("Contribution to Effective Stress",
GetType(Single))  'Column 14
            CalculationsDataTable.Columns.Add("Cumulative Effective Pressure",
GetType(Single))     'Column 15
            CalculationsDataTable.Columns.Add("Pressure From Uniform Loads",
GetType(Single))       'Column 16
            CalculationsDataTable.Columns.Add("Earth Pressure", GetType(Single))
'Column 17
            CalculationsDataTable.Columns.Add("Earth Pressure Adjusted for Tension Crack",
GetType(Single))     'Column 18


            CalculationsDataTable.Rows(0).Item(13) = 0
            CalculationsDataTable.Rows(0).Item(14) =
CalculationsDataTable.Rows(0).Item(12) _
                * CalculationsDataTable.Rows(0).Item(13)
            CalculationsDataTable.Rows(0).Item(15) =
CalculationsDataTable.Rows(0).Item(14)
            CalculationsDataTable.Rows(0).Item(16) = Surcharge

            For i = 1 To CalculationsDataTable.Rows.Count - 1
                CalculationsDataTable.Rows(i).Item(13) =
Abs(CalculationsDataTable.Rows(i).Item(0) - CalculationsDataTable.Rows(i - 1).Item(0))
                CalculationsDataTable.Rows(i).Item(14) =
CalculationsDataTable.Rows(i).Item(12) * CalculationsDataTable.Rows(i).Item(13)
                CalculationsDataTable.Rows(i).Item(15) =
CalculationsDataTable.Rows(i).Item(14) + CalculationsDataTable.Rows(i - 1).Item(15)
                CalculationsDataTable.Rows(i).Item(16) = Surcharge
            Next


            'Calculation of Earth Pressure (Unresolved)
            For Each CalculationRow As DataRow In CalculationsDataTable.Rows
                CalculationRow.Item(17) = CalculationRow.Item(9) * _
                    (CalculationRow.Item(15) + CalculationRow.Item(16)) +
CalculationRow.Item(10)
            Next CalculationRow


            'checking for tension cracks and ajusting the Earth Pressure
            i = 0
            Do Until i = CalculationsDataTable.Rows.Count - 1
                If CalculationsDataTable.Rows(i).Item(17) < 0 Then
                    If CalculationsDataTable.Rows(i + 1).Item(17) <= 0 Then
                        'Both Values within the segment are -ve, change both to zero
                        CalculationsDataTable.Rows(i).Item(18) = 0
                        CalculationsDataTable.Rows(i + 1).Item(18) = 0
```

```
                        If HydrostaticDueToCrackEnabled And ProgressiveTensileCrack Then
_
                            WaterInfiltrationDepth = Min(CalculationsDataTable.Rows(i +
1).Item(0), GroundWaterDepth)
                    Else
                        'Only the top value is -ve, find the point of zero pressure &
introduce a new row
                        Dim ZeroPoint As Single
                        ZeroPoint = CalculationsDataTable.Rows(i).Item(0) +
CalculationsDataTable.Rows(i).Item(17) _
                            / (CalculationsDataTable.Rows(i + 1).Item(17) -
CalculationsDataTable.Rows(i).Item(17)) _
                            * (CalculationsDataTable.Rows(i).Item(0) -
CalculationsDataTable.Rows(i + 1).Item(0))

                        CalculationsDataTable.Rows.Add()

                        For j = CalculationsDataTable.Rows.Count - 2 To i + 1 Step -1
                            For k = 0 To CalculationsDataTable.Columns.Count - 1
                                CalculationsDataTable.Rows(j + 1).Item(k) =
CalculationsDataTable.Rows(j).Item(k)
                            Next
                        Next

                        CalculationsDataTable.Rows(i + 1).Item(0) = Round(ZeroPoint, 3)
                        If GroundWaterDepth >= CalculationsDataTable.Rows(i + 1).Item(0)
Then
                            CalculationsDataTable.Rows(i + 1).Item(11) = 0
                        Else
                            CalculationsDataTable.Rows(i + 1).Item(11) = _
                                CalculationsDataTable.Rows(i + 1).Item(0) -
GroundWaterDepth
                        End If

                        CalculationsDataTable.Rows(i + 1).Item(13) = Abs(ZeroPoint -
CalculationsDataTable.Rows(i).Item(0))
                        CalculationsDataTable.Rows(i + 2).Item(13) =
Abs(CalculationsDataTable.Rows(i + 2).Item(0) - ZeroPoint)
                        CalculationsDataTable.Rows(i + 1).Item(14) =
CalculationsDataTable.Rows(i + 1).Item(12) * CalculationsDataTable.Rows(i + 1).Item(13)
                        CalculationsDataTable.Rows(i + 2).Item(14) =
CalculationsDataTable.Rows(i + 2).Item(12) * CalculationsDataTable.Rows(i + 2).Item(13)
                        CalculationsDataTable.Rows(i + 1).Item(15) =
CalculationsDataTable.Rows(i + 1).Item(14) + CalculationsDataTable.Rows(i).Item(15)
                        CalculationsDataTable.Rows(i + 1).Item(17) = 0
                        CalculationsDataTable.Rows(i).Item(18) = 0
                        CalculationsDataTable.Rows(i + 1).Item(18) = 0

                        If HydrostaticDueToCrackEnabled And ProgressiveTensileCrack Then
                            WaterInfiltrationDepth = Min(Round(ZeroPoint, 3),
GroundWaterDepth)
                            ProgressiveTensileCrack = False
                        End If
                    End If
                Else
                    CalculationsDataTable.Rows(i).Item(18) =
CalculationsDataTable.Rows(i).Item(17)
```

```
                    CalculationsDataTable.Rows(i + 1).Item(18) =
CalculationsDataTable.Rows(i + 1).Item(17)

                    ProgressiveTensileCrack = False
                End If
                i += 1
            Loop

            CalculationsDataTable.Columns.Add("Horizontal Pressure Component of Earth
Pressure", GetType(Single)) 'Column 19
            CalculationsDataTable.Columns.Add("Hydrostatic Pressure due to G.W.T.",
GetType(Single))         'Column 20
            CalculationsDataTable.Columns.Add("Hydrostatic Pressure due to Tension Crack",
GetType(Single))         'Column 21
            CalculationsDataTable.Columns.Add("Total Horizontal Pressure",
GetType(Single))     'Column 22
            CalculationsDataTable.Columns.Add("Horizontal Force From Segment",
GetType(Single))     'Column23
            CalculationsDataTable.Columns.Add("Lever Arm (from the Bottom)",
GetType(Single))   'Column24
            CalculationsDataTable.Columns.Add("Moment from Horizontal Force of Segment",
GetType(Single))    'Column25
            CalculationsDataTable.Columns.Add("Unit Weight of Segment for Vertical Force
on Wall", GetType(Single)) 'Column26
            CalculationsDataTable.Columns.Add("Segment Area for Weight Calculation",
GetType(Single))   'Column27
            CalculationsDataTable.Columns.Add("Weight of Segment", GetType(Single))
'Column28
            CalculationsDataTable.Columns.Add("Lever Arm of Segment", GetType(Single))
'Column29
            CalculationsDataTable.Columns.Add("Moment From Segment", GetType(Single))
'Column30
            CalculationsDataTable.Columns.Add("Moment From Surface Load", GetType(Single))
'Column31
            CalculationsDataTable.Columns.Add("Vertical Component of Earth Pressure",
GetType(Single))   'Column32
            CalculationsDataTable.Columns.Add("Vertical Force from Earth Pressure of
Segment", GetType(Single))     'Column33
            CalculationsDataTable.Columns.Add("Lever Arm of Vertical Force",
GetType(Single))     'Column34
            CalculationsDataTable.Columns.Add("Moment from Vertical Force",
GetType(Single))        'Column35

            For Each CalculationRow As DataRow In CalculationsDataTable.Rows
                'If Theory = "Rankine" Or Theory = "Bell" Then
                If Theory = "Rankine" Then
                    CalculationRow.Item(19) = Cos(BackfillInclinationBeta) *
CalculationRow.Item(18)
                    CalculationRow.Item(32) = Sin(BackfillInclinationBeta) *
CalculationRow.Item(18)
                ElseIf Theory = "Coulomb" Then
                    Dim delta As Single
                    delta = CalculationRow.Item(7)
                    CalculationRow.Item(19) = Cos(delta + WallInclinationAlpha) *
CalculationRow.Item(18)
                    CalculationRow.Item(32) = Sin(delta + WallInclinationAlpha) *
CalculationRow.Item(18)
                ElseIf Theory = "KeriselAbsi" Then
```

```
                    Dim delta As Single
                    delta = CalculationRow.Item(7)
                    CalculationRow.Item(19) = Cos(delta + WallInclinationAlpha) /
Cos(delta) * CalculationRow.Item(18)
                    CalculationRow.Item(32) = Sin(delta + WallInclinationAlpha) /
Cos(delta) * CalculationRow.Item(18)
                Else
                    'At rest earth pressure and Bell's solution for Cohesive/mixed soils
                    CalculationRow.Item(19) = CalculationRow.Item(18)
                    CalculationRow.Item(32) = 0
                End If
            Next

            'Calculation of Hydrostatic Pressure due to GWT (Column 20)
            For Each CalculationRow In CalculationsDataTable.Rows
                If CalculationRow.item(4) = "Effective Stress" Then
                    CalculationRow.Item(20) = CalculationRow.Item(11) * UnitWeightOfWater
                Else
                    CalculationRow.Item(20) = 0
                End If
                CalculationRow.Item(21) = 0
            Next CalculationRow

            'Calculation of Hydrostatic Pressure due to Tension Crack (Column 21)
            For i = 2 To CalculationsDataTable.Rows.Count - 1 Step 1
                If Round(CalculationsDataTable.Rows(i).Item(0), 3) =
Round(WaterInfiltrationDepth, 3) _
                    And i <> CalculationsDataTable.Rows.Count - 1 Then
                    CalculationsDataTable.Rows(i).Item(21) = _
                        UnitWeightOfWater * CalculationsDataTable.Rows(i).Item(0)
                    If Round(CalculationsDataTable.Rows(i + 1).Item(0), 3) =
Round(WaterInfiltrationDepth, 3) Then
                        Exit For
                    Else
                        CalculationsDataTable.Rows.Add()
                        For j = CalculationsDataTable.Rows.Count - 2 To i Step -1
                            For k = 0 To CalculationsDataTable.Columns.Count - 1
                                CalculationsDataTable.Rows(j + 1).Item(k) =
CalculationsDataTable.Rows(j).Item(k)
                            Next
                        Next
                        CalculationsDataTable.Rows(i + 1).Item(21) = 0
                        Exit For
                    End If
                ElseIf CalculationsDataTable.Rows(i).Item(0) <= WaterInfiltrationDepth
Then
                    CalculationsDataTable.Rows(i).Item(21) = _
                        UnitWeightOfWater * CalculationsDataTable.Rows(i).Item(0)
                End If
            Next

            'Calculation of Total Horizontal Pressure (Column 22)
            For Each CalculationRow In CalculationsDataTable.Rows
                CalculationRow.item(22) = CalculationRow.Item(19) +
Max(CalculationRow.Item(20), CalculationRow.item(21))
            Next CalculationRow
```

```
            'Resultant Horizontal Force Calculation
            Dim SumOfHorizontalForces, SumOfHorizontalMoments As Single
            Dim AvgPressure, SegmentThichness, LeverArm As Single
            Dim P1, P2 As Single

            For i = 1 To CalculationsDataTable.Rows.Count - 1
                P1 = CalculationsDataTable.Rows(i - 1).Item(22)
                P2 = CalculationsDataTable.Rows(i).Item(22)
                AvgPressure = 0.5 * (P1 + P2)
                SegmentThichness = CalculationsDataTable.Rows(i).Item(13)

                CalculationsDataTable.Rows(i).Item(23) = AvgPressure * SegmentThichness
                SumOfHorizontalForces += AvgPressure * SegmentThichness

                If P1 = 0 And P2 = 0 Then
                    LeverArm = 0.5 * (CalculationsDataTable.Rows(i - 1).Item(0) +
CalculationsDataTable.Rows(i).Item(0))
                Else
                    LeverArm = 0.5 * (CalculationsDataTable.Rows(i - 1).Item(0) +
CalculationsDataTable.Rows(i).Item(0)) _
                        + SegmentThichness * (P2 - P1) / 6 / (P2 + P1)
                End If
                LeverArm = HeightOfAnalysis - LeverArm
                CalculationsDataTable.Rows(i).Item(24) = LeverArm

                CalculationsDataTable.Rows(i).Item(25) = AvgPressure * SegmentThichness *
LeverArm
                SumOfHorizontalMoments += AvgPressure * SegmentThichness * LeverArm
            Next
            HorizontalResultantForce = SumOfHorizontalForces

            If SumOfHorizontalForces = 0 Then
                HorizontalResultantLocation = HeightOfAnalysis / 2
            Else
                HorizontalResultantLocation = SumOfHorizontalMoments /
SumOfHorizontalForces
            End If

            CalculationParametersTable.Rows.Add()
            RowNumber = CalculationParametersTable.Rows.Count - 1
            CalculationParametersTable.Rows(RowNumber).Item(0) = "Horizontal Force from
Earth Pressure"
            CalculationParametersTable.Rows(RowNumber).Item(1) = "kN"
            CalculationParametersTable.Rows(RowNumber).Item(2) =
Round(SumOfHorizontalForces, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 1).Item(0) = "Moment from
Horizontal Earth Pressure"
            CalculationParametersTable.Rows(RowNumber + 1).Item(1) = "kN.m"
            CalculationParametersTable.Rows(RowNumber + 1).Item(2) =
Round(SumOfHorizontalMoments, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 2).Item(0) = "z-coord. of
Horizontal Resultant"
            CalculationParametersTable.Rows(RowNumber + 2).Item(1) = "m"
            If StabilityAnalysis Then
```

```
                CalculationParametersTable.Rows(RowNumber + 2).Item(2) =
Round(HorizontalResultantLocation, 3)
            Else
                CalculationParametersTable.Rows(RowNumber + 2).Item(2) _
                    = Round(HeightOfAnalysis - HorizontalResultantLocation, 3)
            End If


            'Calculation of the force from the Weight of Wedge and surface load

            Dim SurchargeForce, SurchargeMoment As Single
            Dim WeightOfWedge, MomentFromWedge As Single

            If Round(FLDistance, 3) > 0 Then
                If FrontFacePassiveAnalysis Then
                    SurchargeForce = (FLDistance - S1 * (H - PassiveSoilThickness)) *
Surcharge
                    SurchargeMoment = 0.5 * (FLDistance - S1 * (H - PassiveSoilThickness))
^ 2 * Surcharge
                Else
                    SurchargeForce = CalculationsDataTable.Rows(0).Item(16) _
                        * FLDistance / Cos(BackfillInclinationBeta)
                    SurchargeMoment = SurchargeForce * (0.5 * FLDistance +
HorizontalLever)
                End If
            End If


            For i = 1 To CalculationsDataTable.Rows.Count - 1
                'choosing from bulk unit weight & saturated unit weight
                If CalculationsDataTable.Rows(i).Item(0) <= GroundWaterDepth Then
                    CalculationsDataTable.Rows(i).Item(26) =
CalculationsDataTable.Rows(i).Item(2)
                Else
                    CalculationsDataTable.Rows(i).Item(26) =
CalculationsDataTable.Rows(i).Item(3)
                End If

                If Round(FLDistance, 3) > 0 Then


                    If i = 1 Then
                        'Area, Weight and LeverArm of top triangular wedge
                        CalculationsDataTable.Rows(i).Item(27) = 0.5 * (FLDistance) ^ 2 _
                            * Tan(BackfillInclinationBeta)
                        CalculationsDataTable.Rows(i).Item(28) =
CalculationsDataTable.Rows(i).Item(26) _
                            * CalculationsDataTable.Rows(i).Item(27)
                        CalculationsDataTable.Rows(i).Item(29) = HorizontalLever + 2.0 /
3.0 * FLDistance

                        CalculationsDataTable.Rows(i).Item(30) =
CalculationsDataTable.Rows(i).Item(28) _
                            * CalculationsDataTable.Rows(i).Item(29)

                    ElseIf i > 1 Then
                        Dim s, w As Single
```

```
                        If StabilityAnalysis Then
                            If CalculationsDataTable.Rows(i).Item(0) > (H - d) Then
                                If CalculationsDataTable.Rows(i - 1).Item(0) <= (H - d)
Then
                                    'segment is partly below foundation pad
                                    s = H - d - CalculationsDataTable.Rows(i - 1).Item(0)
                                    w = PadOverhang
                                Else
                                    s = 0
                                    w = 0
                                End If
                            Else
                                'segment is fully above foundation pad
                                s = CalculationsDataTable.Rows(i).Item(13)
                                w = FLDistance - _
                                    CalculationsDataTable.Rows(i).Item(0)  *  Tan(WallAngle)
                            End If
                        Else    'earth pressure analysis problem
                            s = CalculationsDataTable.Rows(i).Item(13)
                            w = FLDistance - CalculationsDataTable.Rows(i).Item(0) _
                                * Tan(WallAngle)
                        End If
                        CalculationsDataTable.Rows(i).Item(27) = 0.5 * s ^ 2 *
Tan(WallAngle) + s * w

                        CalculationsDataTable.Rows(i).Item(28) =
CalculationsDataTable.Rows(i).Item(26) _
                            * CalculationsDataTable.Rows(i).Item(27)

                        CalculationsDataTable.Rows(i).Item(30) = 0.5 * s ^ 2 *
Tan(WallAngle) _
                            * (HorizontalLever + FLDistance - w - 1.0 / 3.0 * s *
Tan(WallAngle)) _
                            + s * w * (HorizontalLever + FLDistance - 0.5 * w)

                        CalculationsDataTable.Rows(i).Item(30) =
CalculationsDataTable.Rows(i).Item(26) _
                            * CalculationsDataTable.Rows(i).Item(30)

                        If CalculationsDataTable.Rows(i).Item(27) = 0 Then
                            CalculationsDataTable.Rows(i).Item(29) = HorizontalLever +
FLDistance - 0.5 * w
                        Else
                            CalculationsDataTable.Rows(i).Item(29) =
CalculationsDataTable.Rows(i).Item(30) _
                                / CalculationsDataTable.Rows(i).Item(28)
                        End If

                        'adjusing for the distance from toe to the centroid of passive
soil
                        If FrontFacePassiveAnalysis Then

                            CalculationsDataTable.Rows(1).Item(29) = 0.5 * (FLDistance -
S1 * (H - PassiveSoilThickness))

                            CalculationsDataTable.Rows(i).Item(29) = FLDistance - _
                                CalculationsDataTable.Rows(i).Item(29)
```

```
                            CalculationsDataTable.Rows(i).Item(30) =
CalculationsDataTable.Rows(i).Item(28) _
                                * CalculationsDataTable.Rows(i).Item(29)
                        End If
                    End If

                    WeightOfWedge += CalculationsDataTable.Rows(i).Item(28)
                    MomentFromWedge += CalculationsDataTable.Rows(i).Item(30)
                End If
            Next

            'If FrontFacePassiveAnalysis = False Then
            CalculationsDataTable.Rows(1).Item(31) = SurchargeMoment

            ForceFromWedge = WeightOfWedge + SurchargeForce
            If ForceFromWedge <> 0 Then
                WedgeLeverArm = (MomentFromWedge + SurchargeMoment) / ForceFromWedge
            Else
                WedgeLeverArm = 0.5 * FLDistance + HorizontalLever
            End If

            CalculationParametersTable.Rows.Add()
            RowNumber = CalculationParametersTable.Rows.Count - 1
            CalculationParametersTable.Rows(RowNumber).Item(0) = "Weight of Soil Wedge"
            CalculationParametersTable.Rows(RowNumber).Item(1) = "kN"
            CalculationParametersTable.Rows(RowNumber).Item(2) = Round(WeightOfWedge, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 1).Item(0) = "Moment from Soil
Wedge"
            CalculationParametersTable.Rows(RowNumber + 1).Item(1) = "kN.m"
            CalculationParametersTable.Rows(RowNumber + 1).Item(2) =
Round(MomentFromWedge, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 2).Item(0) = "Force from Surface
Load"
            CalculationParametersTable.Rows(RowNumber + 2).Item(1) = "kN"
            CalculationParametersTable.Rows(RowNumber + 2).Item(2) = Round(SurchargeForce,
3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 3).Item(0) = "Moment from Surface
Load"
            CalculationParametersTable.Rows(RowNumber + 3).Item(1) = "kN.m"
            CalculationParametersTable.Rows(RowNumber + 3).Item(2) =
Round(SurchargeMoment, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 4).Item(0) = "x-coord. of Wedge
Force"
            CalculationParametersTable.Rows(RowNumber + 4).Item(1) = "m"
            CalculationParametersTable.Rows(RowNumber + 4).Item(2) = Round(WedgeLeverArm,
3)
            'End If

            'Calculation of Vertical Resultant Force
            Dim SumOfVerticalForces, SumOfVerticalMoments As Single
```

```
            For i = 0 To CalculationsDataTable.Rows.Count - 1
                If i = 0 Then
                    CalculationsDataTable.Rows(i).Item(33) = 0
                    CalculationsDataTable.Rows(i).Item(34) = HorizontalLever + FLDistance
                    CalculationsDataTable.Rows(i).Item(35) = 0
                Else
                    If AnalysisType = "At Rest Earth Pressure" Then
                        CalculationsDataTable.Rows(i).Item(33) = 0
                    Else
                        P1 = CalculationsDataTable.Rows(i - 1).Item(32)
                        P2 = CalculationsDataTable.Rows(i).Item(32)
                        AvgPressure = 0.5 * (P1 + P2)
                        CalculationsDataTable.Rows(i).Item(33) = AvgPressure *
CalculationsDataTable.Rows(i).Item(13)
                    End If

                    SumOfVerticalForces += CalculationsDataTable.Rows(i).Item(33)

                    If FLDistance = 0 Then
                        If P1 = 0 And P2 = 0 Then
                            Dim a11, b, c, d As Single
                            LeverArm = (Abs(CalculationsDataTable.Rows(i - 1).Item(0)) _
                                  + 0.5 * CalculationsDataTable.Rows(i).Item(13)) _
                                * Tan(WallAngle) + HorizontalLever

                            a11 = Abs(CalculationsDataTable.Rows(i - 1).Item(0) _
                                    - CalculationsDataTable.Rows(1).Item(0))
                            b = 0.5 * CalculationsDataTable.Rows(i).Item(13)
                            c = CalculationsDataTable.Rows(i).Item(13)
                            d = HorizontalLever

                        Else

                            LeverArm = (Abs(CalculationsDataTable.Rows(i - 1).Item(0)) _
                                  + 0.5 * CalculationsDataTable.Rows(i).Item(13) _
                                  + CalculationsDataTable.Rows(i).Item(13) * (P2 -
P1) / 6 / (P2 + P1)) _
                                * Tan(WallAngle) + HorizontalLever
                            Dim a11, b, c, d As Single
                            a11 = Abs(CalculationsDataTable.Rows(i - 1).Item(0) _
                                    - CalculationsDataTable.Rows(1).Item(0))
                            b = 0.5 * CalculationsDataTable.Rows(i).Item(13)
                            c = CalculationsDataTable.Rows(i).Item(13) * (P2 - P1) / 6 /
(P2 + P1)
                            d = HorizontalLever

                        End If
                        If FrontFacePassiveAnalysis Then LeverArm = PadOverhang + (H - d)
* Tan(WallAngle) - LeverArm
                    Else
                        LeverArm = HorizontalLever + FLDistance
                        If FrontFacePassiveAnalysis Then LeverArm = 0
                    End If

                    CalculationsDataTable.Rows(i).Item(34) = LeverArm

                    CalculationsDataTable.Rows(i).Item(35) = AvgPressure * _
                        CalculationsDataTable.Rows(i).Item(13) * LeverArm
```

```
                    SumOfVerticalMoments += AvgPressure *
CalculationsDataTable.Rows(i).Item(13) * LeverArm
                End If
            Next
            VerticalResultantForce = SumOfVerticalForces

            If SumOfVerticalForces = 0 Then
                VerticalResultantLocation = HorizontalLever + FLDistance _
                    + 0.5 * HeightOfAnalysis * Tan(WallInclinationAlpha)
            Else
                VerticalResultantLocation = SumOfVerticalMoments / SumOfVerticalForces
            End If

            CalculationParametersTable.Rows.Add()
            RowNumber = CalculationParametersTable.Rows.Count - 1
            CalculationParametersTable.Rows(RowNumber).Item(0) = "Vertical Force from
Earth Pressure"
            CalculationParametersTable.Rows(RowNumber).Item(1) = "kN"
            CalculationParametersTable.Rows(RowNumber).Item(2) =
Round(SumOfVerticalForces, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 1).Item(0) = "Moment from
Vertical Earth Pressure"
            CalculationParametersTable.Rows(RowNumber + 1).Item(1) = "kN.m"
            CalculationParametersTable.Rows(RowNumber + 1).Item(2) =
Round(SumOfVerticalMoments, 3)

            CalculationParametersTable.Rows.Add()
            CalculationParametersTable.Rows(RowNumber + 2).Item(0) = "x-coord. of
Vertical Resultant"
            CalculationParametersTable.Rows(RowNumber + 2).Item(1) = "m"
            CalculationParametersTable.Rows(RowNumber + 2).Item(2) =
Round(VerticalResultantLocation, 3)


            For Each CalculationRow In CalculationsDataTable.Rows
                CalculationRow.item(5) = Round(CalculationRow.item(5) * 180 / Math.PI, 3)
                Try
                    CalculationRow.item(7) = Round(CalculationRow.item(7) * 180 / Math.PI,
3)
                Catch ex As Exception
                End Try

                If StabilityAnalysis Then _
                    CalculationRow.Item(0) = HeightOfAnalysis - CalculationRow.Item(0)
            Next CalculationRow

            'removing the top row if wall and ground inclinations are zero
            If BackfillInclinationBeta = 0 _
                Or FLDistance = 0 _
                Then CalculationsDataTable.Rows.RemoveAt(0)

        Catch
            EarthPressureCalculationError = True
            MsgBox("Error! Input data is either incomplete or contains invalid value")
        End Try
    End Sub
```

```
    Private Sub MainForm_ResizeEnd(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.ResizeEnd
        Refresh()
    End Sub

    Private Sub HeightsGridView_CellValueChanged(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles HeightsGridView.CellValueChanged
        Try
            If UserWallDimensioning Then
                CollectWallDimensions()

                EraseResults()
                Refresh()
            End If
        Catch ex As Exception
        End Try
    End Sub

    Private Sub WidthsGridView_CellValueChanged(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles WidthsGridView.CellValueChanged

        Try
            If UserWallDimensioning Then
                CollectWallDimensions()

                EraseResults()
                Refresh()
            End If
        Catch ex As Exception
        End Try

    End Sub

    Private Sub CollectWallDimensions()

        Try
            'H = HeightsGridView.Rows(0).Cells(0).Value
            If HeightsGridView.Rows(0).Cells(0).Value < H Then
                d = HeightsGridView.Rows(0).Cells(0).Value
            Else
                MsgBox("d has to be less than H")
                UserWallDimensioning = False
                HeightsGridView.Rows(0).Cells(0).Value = d
                UserWallDimensioning = True
            End If

            S1 = HeightsGridView.Rows(1).Cells(0).Value
            S2 = HeightsGridView.Rows(2).Cells(0).Value

            b1 = WidthsGridView.Rows(0).Cells(0).Value
            b2 = WidthsGridView.Rows(1).Cells(0).Value
            b3 = WidthsGridView.Rows(2).Cells(0).Value

            b = b1 + S1 * (H - d) + b2 + S2 * (H - d) + b3

            InitialRetainingDimensions = False
```

```
            ArrangeSoilPropertiesForRetWall()
        Catch ex As Exception
        End Try
    End Sub

    Private Sub StabilityReportButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles StabilityReportButton.Click
        My.Forms.StabilityReport.Show()
    End Sub

    Private Sub CheckStabilityButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CheckStabilityButton.Click
        Dim i As Integer

        'Checking that soil type has been assigned to each layer prior to analysis
        For i = 0 To AssignDataGridView.Rows.Count - 1
            If AssignDataGridView.Rows(i).Cells(2).Value = "" Then
                MsgBox("Assignment of Soil Types for the profile is incomplete.")
                Exit Sub
            End If
        Next

        Dim TempoAnalysisType As String = AnalysisType
        Dim TempoWallInclinatin As Single = WallInclinationAlpha
        Dim TempoFLDistance As Single = FLDistance
        Dim TempoGroundWaterLevel As Single = GroundWaterDepth
        Dim TempoBeta As Single = BackfillInclinationBeta
        Dim TempoSurcharge As Single = Surcharge
        Dim TempowallAngle As Single = WallAngle
        Dim RowNumber As UInteger

        StabilityAnalysis = True

        'Arranging the interfaces necessary for active pressure calculations
        CalculationsInterfaceTable.Rows.Clear()
        For i = 0 To GivenInterfaceTable.Rows.Count - 1
            CalculationsInterfaceTable.Rows.Add()
            CalculationsInterfaceTable.Rows(i).Item(0) =
GivenInterfaceTable.Rows(i).Item(0)
        Next

        'Checking whether fictitious line is needed or not
        TotalStressExists = False
        For i = 0 To AssignmentTable.Rows.Count - 1
            If EpSoilsTable.Rows(AssignmentTable.Rows(i).Item(0)).Item("Stress State") =
"Total Stress" Then TotalStressExists = True
        Next

        If Theory = "Rankine" Or Theory = "Bell" Or Theory = "KeriselAbsi" _
            Or (b3 - 0.15) > 0.00001 Or TotalStressExists Then

            FLDistance = S2 * (H - d) + b3
            WallInclinationAlpha = 0
        Else
            FLDistance = 0
            WallInclinationAlpha = Atan(S2)
        End If
```

```
HorizontalLever = b1 + S1 * (H - d) + b2
PadOverhang = b3
WallAngle = Atan(S2)

AnalysisType = "Active Earth Pressure"

CalculationsDataTable.Columns.Clear()
CalculationsDataTable.Rows.Clear()
CalculateSoilsPropertiesTable = ActiveSoilsTable.Copy

EarthPressureCalculationError = False
CalculateEarthPressure()
If EarthPressureCalculationError Then Exit Sub

ActivePressureTable = CalculationsDataTable.Copy()
WallPressureParametersTable = CalculationParametersTable.Copy

'Creating the Rows of the Forces Table
ResultForcesTable.Rows.Clear()
For i = 0 To 4
    ResultForcesTable.Rows.Add()
Next

ResultForcesTable.Rows(3).Item(0) = "Active Pressure"
ResultForcesTable.Rows(3).Item(1) = HorizontalResultantForce
ResultForcesTable.Rows(3).Item(2) = VerticalResultantForce
ResultForcesTable.Rows(3).Item(3) = VerticalResultantLocation
ResultForcesTable.Rows(3).Item(4) = HorizontalResultantLocation

ResultForcesTable.Rows(2).Item(0) = "Force from Wedge"
ResultForcesTable.Rows(2).Item(2) = ForceFromWedge
ResultForcesTable.Rows(2).Item(3) = WedgeLeverArm

If FLDistance = 0 Then
    'Fictitious line case, no adjustment needed
    GraphicLever = HorizontalResultantLocation
Else
    Dim HydroForce, HydroLever, HydroMoment As Single
    HydroForce = 0.5 * UnitWeightOfWater * (H - GroundWaterDepth)
    HydroLever = (H - GroundWaterDepth) / 3.0
    HydroMoment = HydroLever * HydroForce
    If HorizontalResultantForce > HydroForce Then
        'matching the location of horizontal & vertical forces
        GraphicLever = (HorizontalResultantForce * HorizontalResultantLocation _
                      - HydroMoment) / (HorizontalResultantForce - HydroForce)
    Else
        'HorizontalResultantForce=HydrostaticForce
        GraphicLever = HydroLever
    End If
End If


If PassiveSoilThickness > 0 Then
    FrontFacePassiveAnalysis = True
    GroundWaterDepth = H - PassiveSoilThickness + GroundWaterDepthFront
    Surcharge = PassiveSurcharge
    CalculateSoilsPropertiesTable = PassiveSoilTable.Copy
```

```
            AnalysisType = "Passive Earth Pressure"
            BackfillInclinationBeta = 0

            TotalStressExists = False
            If EpSoilsTable.Rows(PassiveSoilType).Item("Stress State") = "Total Stress"
Then TotalStressExists = True

            If Theory = "Rankine" Or Theory = "Bell" Or Theory = "KeriselAbsi" _
                Or (b1 - 0.15) > 0.00001 Or TotalStressExists Then

                FLDistance = S1 * (H - d) + b1
                WallInclinationAlpha = 0
            Else
                FLDistance = 0
                WallInclinationAlpha = Atan(S1)
            End If
            HorizontalLever = 0
            PadOverhang = b1
            WallAngle = Atan(S1)

            CalculationsInterfaceTable.Rows.Clear()
            CalculationsInterfaceTable.Rows.Add()
            CalculationsInterfaceTable.Rows.Add()
            CalculationsInterfaceTable.Rows(0).Item(0) = H - PassiveSoilThickness
            CalculationsInterfaceTable.Rows(1).Item(0) = H

            CalculationsDataTable.Columns.Clear()
            CalculationsDataTable.Rows.Clear()

            EarthPressureCalculationError = False
            CalculateEarthPressure()
            If EarthPressureCalculationError Then Exit Sub

            If VerticalResultantForce = 0 Then
                VerticalResultantLocation = S1 * (H - d) + b1 - VerticalResultantLocation
            End If

            FrontFacePassiveAnalysis = False

            WallPressureParametersTable.Rows.Add()

            For i = 6 To CalculationParametersTable.Rows.Count - 1
                WallPressureParametersTable.Rows.Add()
                RowNumber = WallPressureParametersTable.Rows.Count - 1
                WallPressureParametersTable.Rows(RowNumber).Item(0) =
CalculationParametersTable.Rows(i).Item(0)
                WallPressureParametersTable.Rows(RowNumber).Item(1) =
CalculationParametersTable.Rows(i).Item(1)
                WallPressureParametersTable.Rows(RowNumber).Item(2) =
CalculationParametersTable.Rows(i).Item(2)
            Next

            PassivePressureTable = CalculationsDataTable.Copy()
            PassivePressureTable.Columns.RemoveAt(21)
            PassivePressureTable.Columns.RemoveAt(18)

            ResultForcesTable.Rows(1).Item(0) = "Front Face Resistance"
            ResultForcesTable.Rows(1).Item(1) = HorizontalResultantForce
```

```
            ResultForcesTable.Rows(1).Item(2) = VerticalResultantForce
            ResultForcesTable.Rows(1).Item(3) = VerticalResultantLocation
            ResultForcesTable.Rows(1).Item(4) = HorizontalResultantLocation

            ResultForcesTable.Rows(4).Item(0) = "Weight of Soil on Front Face"
            ResultForcesTable.Rows(4).Item(2) = ForceFromWedge
            ResultForcesTable.Rows(4).Item(3) = WedgeLeverArm
        Else
            ResultForcesTable.Rows(1).Item(0) = "Front Face Resistance"
            ResultForcesTable.Rows(1).Item(1) = 0
            ResultForcesTable.Rows(1).Item(2) = 0
            ResultForcesTable.Rows(1).Item(3) = 0
            ResultForcesTable.Rows(1).Item(4) = 0

            ResultForcesTable.Rows(4).Item(0) = "Weight of Soil on Front Face"
            ResultForcesTable.Rows(4).Item(2) = 0
            ResultForcesTable.Rows(4).Item(3) = 0
        End If

        'returning earth pressur analysis parameters to the respective variables
        AnalysisType = TempoAnalysisType
        WallInclinationAlpha = TempoWallInclinatin
        FLDistance = TempoFLDistance
        GroundWaterDepth = TempoGroundWaterLevel
        BackfillInclinationBeta = TempoBeta
        Surcharge = TempoSurcharge
        WallAngle = TempowallAngle

        StabilityAnalysis = False


        WallArea = (b * d + 0.5 * S1 * (H - d) ^ 2 + b2 * (H - d) + 0.5 * S2 * (H - d) ^ 
2)
        WallWeight = GammaRtWall * WallArea
        WallMoment = ((b * d) * (0.5 * b) _
                                    + (0.5 * S1 * (H - d) ^ 2) * (b1 + S1 * (H - d) * 2.0 
/ 3.0) _
                                    + (b2 * (H - d)) * (b1 + S1 * (H - d) + b2 / 2.0) _
                                    + (0.5 * S2 * (H - d) ^ 2) * (b1 + S1 * (H - d) + b2 
+ S2 * (H - d) / 3))

        'To avoid the 0/0 case, wall lever arm is calculated using area of wall
        WallLeverArm = WallMoment / WallArea
        WallMoment = GammaRtWall * WallMoment

        ResultForcesTable.Rows(0).Item(0) = "Weight of Wall"
        ResultForcesTable.Rows(0).Item(2) = WallWeight
        ResultForcesTable.Rows(0).Item(3) = WallLeverArm


        'Checking overturning
        Dim ResistingMoment, OverturningMoment As Single
        Dim OverturningFS As Single

        StabilityParametersTable.Rows.Clear()

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
```

```
        StabilityParametersTable.Rows(1).Item(0) = "1. Check for Overturning"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(3).Item(0) = "Resisting Moments"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(5).Item(0) = "Unit Wt. of Wall"
        StabilityParametersTable.Rows(5).Item(1) = "kN/m3"
        StabilityParametersTable.Rows(5).Item(2) = GammaRtWall

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(6).Item(0) = "Area of Wall"
        StabilityParametersTable.Rows(6).Item(1) = "m2"
        StabilityParametersTable.Rows(6).Item(2) = WallArea

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(7).Item(0) = "Weight of Wall"
        StabilityParametersTable.Rows(7).Item(1) = "kN"
        StabilityParametersTable.Rows(7).Item(2) = WallWeight

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(8).Item(0) = "x-coord. of Wall Weight"
        StabilityParametersTable.Rows(8).Item(1) = "m"
        StabilityParametersTable.Rows(8).Item(2) = WallLeverArm

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(9).Item(0) = "Moment from Wall Weight"
        StabilityParametersTable.Rows(9).Item(1) = "kN.m"
        StabilityParametersTable.Rows(9).Item(2) = WallMoment

        ResistingMoment += StabilityParametersTable.Rows(9).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Weight of Retained Soil
Wedge"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber).Item(2) =
ResultForcesTable.Rows(2).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Lever Arm of Retained
Soil Wedge"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "m"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) =
ResultForcesTable.Rows(2).Item(3)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Moment from Retained Soil
Wedge"
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) =
ResultForcesTable.Rows(2).Item(2) _
            * ResultForcesTable.Rows(2).Item(3)
```

```
        ResistingMoment += StabilityParametersTable.Rows(RowNumber + 2).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Active Earth Pressure:
Vertical Component"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber).Item(2) =
ResultForcesTable.Rows(3).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Lever Arm of Active
Vertical Pressure"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "m"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) =
ResultForcesTable.Rows(3).Item(3)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Moment from Active
Vertical Pressure"
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) =
ResultForcesTable.Rows(3).Item(2) _
            * ResultForcesTable.Rows(3).Item(3)

        ResistingMoment += StabilityParametersTable.Rows(RowNumber + 2).Item(2)


        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Weight of Soil on Front Face"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber).Item(2) =
ResultForcesTable.Rows(4).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Lever Arm of Soil on
Front Face"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "m"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) =
ResultForcesTable.Rows(4).Item(3)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Moment from Weight of
Front Face Soil"
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) =
ResultForcesTable.Rows(4).Item(2) _
            * ResultForcesTable.Rows(4).Item(3)

        ResistingMoment += StabilityParametersTable.Rows(RowNumber + 2).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Resisting Moment"
```

```
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber).Item(2) = ResistingMoment

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Overturning Moment"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Active Earth Pressure:
Horizontal Component"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber).Item(2) =
ResultForcesTable.Rows(3).Item(1)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Lever Arm of Horizontal
Active Pressure"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "m"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) =
ResultForcesTable.Rows(3).Item(4)

        OverturningMoment = StabilityParametersTable.Rows(RowNumber).Item(2) _
            * StabilityParametersTable.Rows(RowNumber + 1).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Moment from Horizontal
Active Pressure"
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) = OverturningMoment

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Overturning Moment"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber).Item(2) = OverturningMoment

        OverturningFS = Round(ResistingMoment / OverturningMoment, 2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "F.S. against Overturning"
        StabilityParametersTable.Rows(RowNumber).Item(2) = OverturningFS


        'Check Sliding
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "2. Check for Sliding"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
```

```
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Vertical Reaction Components"

        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Weight of Retained Soil
Wedge"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber).Item(2) =
ResultForcesTable.Rows(2).Item(2)
        VerticalReaction = StabilityParametersTable.Rows(RowNumber).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Weight of Wall"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) = WallWeight
        VerticalReaction += WallWeight

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Weight of Soil on Front
Face"
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) =
ResultForcesTable.Rows(4).Item(2)
        VerticalReaction += StabilityParametersTable.Rows(RowNumber + 2).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 3).Item(0) = "Active Earth Pressure:
Vertical Component"
        StabilityParametersTable.Rows(RowNumber + 3).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber + 3).Item(2) =
ResultForcesTable.Rows(3).Item(2)
        VerticalReaction += StabilityParametersTable.Rows(RowNumber + 3).Item(2)

        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Total Vertical Reaction"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber).Item(2) = VerticalReaction

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Resisting Force Components"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Foundation Friction"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) = VerticalReaction *
Tan(FoundationFriction)
        AntiSlidingForce = StabilityParametersTable.Rows(RowNumber + 1).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Foundation Adhesion"
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) = FoundationAdhesion * b
        AntiSlidingForce += StabilityParametersTable.Rows(RowNumber + 2).Item(2)

        StabilityParametersTable.Rows.Add()
```

```
        StabilityParametersTable.Rows(RowNumber + 3).Item(0) = "Passive Earth Pressure:
Horizontal Component"
        StabilityParametersTable.Rows(RowNumber + 3).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber + 3).Item(2) =
ResultForcesTable.Rows(1).Item(1)
        AntiSlidingForce += StabilityParametersTable.Rows(RowNumber + 3).Item(2)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Total Resisting Force"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber).Item(2) = AntiSlidingForce

        Dim SlidingFS As Single
        Dim SlidingForce As Single
        SlidingForce = ResultForcesTable.Rows(3).Item(1)
        SlidingFS = Round((AntiSlidingForce / SlidingForce), 3)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Sliding Force"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Active Earth Pressure:
Horizontal Component"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "kN"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) =
ResultForcesTable.Rows(3).Item(1)


        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "F.S. against Sliding"
        StabilityParametersTable.Rows(RowNumber).Item(2) = SlidingFS


        'Check bearing capacity (Eccentricity & Maximum stress)
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "3. Check for Eccentricity"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "x-coordinate of Vertical
Reaction"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Resisting Moment"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) = ResistingMoment

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Overturning Moment"
```

```
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "kN.m"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) = OverturningMoment

        ReactionLocation = (ResistingMoment - OverturningMoment) / VerticalReaction
        eccentricity = Round(Abs(b / 2 - ReactionLocation), 3)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 3).Item(0) = "x-coordinate of Vertical
Reaction"
        StabilityParametersTable.Rows(RowNumber + 3).Item(1) = "m"
        StabilityParametersTable.Rows(RowNumber + 3).Item(2) = ReactionLocation

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 4).Item(0) = "Eccentricity"
        StabilityParametersTable.Rows(RowNumber + 4).Item(1) = "m"
        'StabilityParametersTable.Rows(RowNumber + 4).Item(2) = eccentricity
        StabilityParametersTable.Rows(RowNumber + 4).Item(2) = eccentricity


        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Check for Eccentricity Limit"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Eccentricity of Ground
Reaction"
        StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "m"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) = eccentricity

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Limit on Eccentricity"
        StabilityParametersTable.Rows(RowNumber + 2).Item(1) = "m"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) = Round(b / 6, 3)

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "4. Check Maximum & Minimum
Stresses"

        If BearingCapacityByUser = False Then CalculateBearingCapacity()
        AllowableBearingCapacity = UltimateBearingCapacity / BearingCapacityFS

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Allowable Bearing Capacity"
        StabilityParametersTable.Rows(RowNumber).Item(1) = "kPa"
        StabilityParametersTable.Rows(RowNumber).Item(2)  =  Round(AllowableBearingCapacity,
3)

        Dim QMax, QMin As Single
        If eccentricity > 0 Then
            QMax = VerticalReaction / b * (1 + 6 * eccentricity / b)
            QMin = VerticalReaction / b * (1 - 6 * eccentricity / b)
        Else
            QMax = VerticalReaction / b * (1 - 6 * eccentricity / b)
```

```
        QMin = VerticalReaction / b * (1 + 6 * eccentricity / b)
    End If

    StabilityParametersTable.Rows.Add()
    StabilityParametersTable.Rows.Add()
    RowNumber = StabilityParametersTable.Rows.Count - 1
    StabilityParametersTable.Rows(RowNumber).Item(0) = "Maximum Stress"
    StabilityParametersTable.Rows(RowNumber).Item(1) = "kPa"
    StabilityParametersTable.Rows(RowNumber).Item(2) = Round(QMax, 3)

    StabilityParametersTable.Rows.Add()
    StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Minimum Stress"
    StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "kPa"
    StabilityParametersTable.Rows(RowNumber + 1).Item(2) = Round(QMin, 3)


    'Preparing the stability analysis table
    SafetyFactorTable.Rows.Clear()

    'Factors of Safety Table: Row 0
    SafetyFactorTable.Rows.Add()
    SafetyFactorTable.Rows(0).Item(0) = "Overturning, Factor of Safety"
    SafetyFactorTable.Rows(0).Item(1) = "> " & CStr(AllowedOverturnFS)
    SafetyFactorTable.Rows(0).Item(2) = OverturningFS
    If OverturningFS >= AllowedOverturnFS Then
        SafetyFactorTable.Rows(0).Item(3) = "Pass"
    Else
        SafetyFactorTable.Rows(0).Item(3) = "Fail"
    End If

    'Factors of Safety Table: Row 1
    SafetyFactorTable.Rows.Add()
    SafetyFactorTable.Rows(1).Item(0) = "Sliding, Factor of Safety"
    SafetyFactorTable.Rows(1).Item(1) = "> " & CStr(AllowedSlipFS)
    SafetyFactorTable.Rows(1).Item(2) = SlidingFS
    If SlidingFS >= AllowedSlipFS Then
        SafetyFactorTable.Rows(1).Item(3) = "Pass"
    Else
        SafetyFactorTable.Rows(1).Item(3) = "Fail"
    End If

    'Factors of Safety Table: Row 2
    SafetyFactorTable.Rows.Add()
    SafetyFactorTable.Rows(2).Item(0) = "Bearing Capacity, Eccentricity"
    SafetyFactorTable.Rows(2).Item(1) = "< " & CStr(Round(b / 6.0, 3))
    SafetyFactorTable.Rows(2).Item(2) = Abs(eccentricity)
    If (b / 6.0) >= Abs(eccentricity) Then
        SafetyFactorTable.Rows(2).Item(3) = "Pass"
    Else
        SafetyFactorTable.Rows(2).Item(3) = "Fail"
    End If

    'Factors of Safety Table: Row 3
    SafetyFactorTable.Rows.Add()
    SafetyFactorTable.Rows(3).Item(0) = "Bearing Capacity, Max-Stress"
    SafetyFactorTable.Rows(3).Item(1) = "< " & CStr(AllowableBearingCapacity)
    SafetyFactorTable.Rows(3).Item(2) = QMax
    If AllowableBearingCapacity >= QMax Then
```

```
            SafetyFactorTable.Rows(3).Item(3) = "Pass"
        Else
            SafetyFactorTable.Rows(3).Item(3) = "Fail"
        End If


        RetOutputTable.Columns.Clear()
        RetOutputTable.Rows.Clear()
        RetOutputTable.Columns.Add("Depth")
        RetOutputTable.Columns.Add("Horizontal Component", GetType(Single))
        RetOutputTable.Columns.Add("Vertical Component", GetType(Single))
        Dim j As Integer = 1

        RetOutputTable.Rows.Add()
        RetOutputTable.Rows(0).Item(0) = "Active"
        For i = 0 To ActivePressureTable.Rows.Count - 1
            RetOutputTable.Rows.Add()
            RetOutputTable.Rows(j).Item(0)  = Round(ActivePressureTable.Rows(i).Item(0), 3)
            RetOutputTable.Rows(j).Item(1) = Round(ActivePressureTable.Rows(i).Item(22),
3)
            RetOutputTable.Rows(j).Item(2) = Round(ActivePressureTable.Rows(i).Item(32),
3)
            j += 1
        Next
        If PassiveSoilThickness <> 0 Then
            RetOutputTable.Rows.Add()
            RetOutputTable.Rows.Add()
            RetOutputTable.Rows(j + 1).Item(0) = "Passive"
            j += 2
            For i = 0 To PassivePressureTable.Rows.Count - 1
                RetOutputTable.Rows.Add()
                RetOutputTable.Rows(j).Item(0) =
Round(PassivePressureTable.Rows(i).Item(0), 3)
                RetOutputTable.Rows(j).Item(1) =
Round(PassivePressureTable.Rows(i).Item(22), 3)
                RetOutputTable.Rows(j).Item(2) =
Round(PassivePressureTable.Rows(i).Item(32), 3)
                j += 1
            Next
        End If

        RetainingWallCalculated = True
        StabilityReportButton.Enabled = True

        'RetainingWallDiagram = True
        OutputGridView.DataSource = RetOutputTable
        OutputGridView.Visible = True
        Refresh()




        'Rounding to 3 decimal places
        For Each DataRow In WallPressureParametersTable.Rows
            Try
                DataRow.item(2) = Round(DataRow.item(2), 3)
            Catch ex As Exception
            End Try
```

```
        Next DataRow



        For Each DataRow In ActivePressureTable.Rows
            For i = 0 To ActivePressureTable.Columns.Count - 1
                Try
                    DataRow.item(i) = Round(DataRow.item(i), 3)
                Catch ex As Exception
                End Try
            Next
        Next DataRow

        For Each DataRow In PassivePressureTable.Rows
            For i = 0 To PassivePressureTable.Columns.Count - 1
                Try
                    DataRow.item(i) = Round(DataRow.item(i), 3)
                Catch ex As Exception
                End Try
            Next
        Next DataRow

        'Rounding to 3 decimal places(Tables on Stability analysis report)
        For Each DataRow In StabilityParametersTable.Rows
            Try
                DataRow.item(2) = Round(DataRow.item(2), 3)
            Catch ex As Exception
            End Try
        Next DataRow

        For Each DataRow In ResultForcesTable.Rows
            For i = 0 To ResultForcesTable.Columns.Count - 1
                Try
                    DataRow.item(i) = Round(DataRow.item(i), 3)
                Catch ex As Exception
                End Try
            Next
        Next DataRow

        For Each DataRow In SafetyFactorTable.Rows
            Try
                DataRow.item(2) = Round(DataRow.item(2), 3)
            Catch ex As Exception
            End Try
        Next DataRow
    End Sub

    Private Sub RefreshRetainingGridViews()
        'HeightsGridView.Rows(0).Cells(0).Value = H
        HeightsGridView.Rows(0).Cells(0).Value = d
        HeightsGridView.Rows(1).Cells(0).Value = S1
        HeightsGridView.Rows(2).Cells(0).Value = S2

        WidthsGridView.Rows(0).Cells(0).Value = b1
        WidthsGridView.Rows(1).Cells(0).Value = b2
        WidthsGridView.Rows(2).Cells(0).Value = b3
    End Sub
```

```
    Private Sub SelectWallType()
        If GravityWallRadioButton.Checked Then
            GravityWallSelected()
        Else
            WallType = "Cantilever"
            H = HeightOfAnalysis
            d = Round(HeightOfAnalysis / 12, 3)
            S1 = Round(1.0 / 48, 3)
            S2 = 0

            b1 = Round(2.0 * H / 15.0, 3)
            b2 = 0.2
            b3 = Max(0, Round(4.0 / 15.0 * H - 11.0 / 12.0 * H * S1 - 0.2, 3))
            b = b1 + S1 * (H - d) + b2 + S2 * (H - d) + b3
        End If
    End Sub

    Private Sub CantileverWallRadioButton_CheckedChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles CantileverWallRadioButton.CheckedChanged
        UserWallDimensioning = False
        SelectWallType()
        RefreshRetainingGridViews()
        UserWallDimensioning = True
        EraseResults()
        Refresh()
    End Sub

    Private Sub GravityWallSelected()
        'Gravity wall is selected 1. Upon loading the main form, and 2. Optionlly by the
user
        WallType = "Gravity"
        H = HeightOfAnalysis
        d = Round(HeightOfAnalysis / 8.0, 3)
        S1 = 1.0 / 50.0
        S2 = 0

        b1 = Round(H / 16.0, 3)
        b2 = Round(H / 12.0, 3)
        b3 = Round(0.3542 * H - 0.875 * H * S1, 3)
        b = b1 + S1 * (H - d) + b2 + S2 * (H - d) + b3
    End Sub

    Private Sub EditButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles EditButton.Click
        Dim i, InterfaceNumber As Integer

        InterfaceNumber = InterfaceDataGridVeiw.Rows.Count
        For i = 0 To InterfaceDataGridVeiw.Rows.Count - 1
            If InterfaceDataGridVeiw.Rows(i).Selected Then InterfaceNumber = i
        Next

        If InterfaceNumber = 0 Then
            MsgBox("The top interface cannot be edited.")
        ElseIf InterfaceNumber = InterfaceDataGridVeiw.Rows.Count - 1 Then
            MsgBox("The bottom interface cannot be edited.")
        ElseIf InterfaceNumber = InterfaceDataGridVeiw.Rows.Count Then
            MsgBox("Please selecte an interface to be edited.")
        Else
```

```
            InterfaceInput.Text = "Edit Interface"
            InterfaceInput.TextBox1.Text =
GivenInterfaceTable.Rows(InterfaceNumber).Item(0)

            My.Forms.InterfaceInput.ShowDialog()

            If InterfaceInput.DialogResult = DialogResult.OK Then
                If NewInterfaceDepth < 0 Or _
    NewInterfaceDepth > HeightOfAnalysis Then Exit Sub


                GivenInterfaceTable.Rows.RemoveAt(InterfaceNumber)

                For i = 0 To GivenInterfaceTable.Rows.Count - 1
                    If GivenInterfaceTable.Rows(i).Item(0) = NewInterfaceDepth Then
                        Refresh()
                        InterfaceDataGridVeiw.Rows(i).Cells(0).Selected = True
                        CreateNewAssignSoilsGridView()
                        Exit Sub
                    End If
                Next

                GivenInterfaceTable.Rows.Add()

                'inserting the new interface depth at the proper location of the sequence
                i = GivenInterfaceTable.Rows.Count - 1
                Do Until GivenInterfaceTable.Rows(i - 1).Item(0) < NewInterfaceDepth
                    GivenInterfaceTable.Rows(i).Item(0) = GivenInterfaceTable.Rows(i -
1).Item(0)

                    i = i - 1
                Loop
                GivenInterfaceTable.Rows(i).Item(0) = NewInterfaceDepth
                InterfaceDataGridVeiw.ClearSelection()
                InterfaceDataGridVeiw.Rows(i).Cells(0).Selected = True

                'editing the Assign Soil table based on the extent of editing conducted
                If i = InterfaceNumber Then
                    'editing hasn't caused the interface to move above or below other
interfaces

                    RefreshAssignTexts()
                Else
                    CreateNewAssignSoilsGridView()
                End If
            End If
        End If

        EraseResults()
        Refresh()
    End Sub

    Private Sub RemoveButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RemoveButton.Click
        Dim i, InterfaceNumber As Integer

        InterfaceNumber = InterfaceDataGridVeiw.Rows.Count
        For i = 0 To InterfaceDataGridVeiw.Rows.Count - 1
            If InterfaceDataGridVeiw.Rows(i).Selected Then InterfaceNumber = i
        Next
```

```
        If InterfaceNumber = 0 Then
            MsgBox("The top interface cannot be removed.")
        ElseIf InterfaceNumber = InterfaceDataGridVeiw.Rows.Count - 1 Then
            MsgBox("The bottom interface cannot be removed.")
        ElseIf InterfaceNumber = InterfaceDataGridVeiw.Rows.Count Then
            MsgBox("Please selecte an interface to be removed.")
        Else
            GivenInterfaceTable.Rows.RemoveAt(InterfaceNumber)
            CreateNewAssignSoilsGridView()
            EraseResults()
            Refresh()
        End If

    End Sub

    Private Sub RemoveSoilsButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RemoveSoilsButton.Click
        Dim i, SoilNumber As Integer

        If SoilDisplayTable.Rows.Count = 0 Then
            MsgBox("There is no soil type to be removed.")
            Exit Sub
        End If

        SoilNumber = SoilDisplayTable.Rows.Count
        For i = 0 To SoilDisplayTable.Rows.Count - 1 Step 1
            If SoilsDataGridView.Rows(i).Selected Then SoilNumber = i
        Next
        If SoilNumber = SoilDisplayTable.Rows.Count Then
            MsgBox("Please select the soil type to be removed.")
            Exit Sub
        End If

        Dim Msg = "Are you sure you want to remove the selected soil type?"
        Dim Result = MsgBox(Msg, 4)
        If Result = 7 Then Exit Sub

        My.Forms.SoilRemovedWarningForm.Show()
        'MsgBox("Soil types have to be assigned to the profiles like new.")

        SoilDisplayTable.Rows.RemoveAt(SoilNumber)
        EpSoilsTable.Rows.RemoveAt(SoilNumber)

        CreateNewAssignSoilsGridView()
        SoilTypeColumn.Items.Clear()
        RetainingWallParameters.PassiveSoilTypeComboBox.Items.Clear()

        For i = 0 To EpSoilsTable.Rows.Count - 1
            SoilTypeColumn.Items.Add(EpSoilsTable.Rows(i).Item(0))

RetainingWallParameters.PassiveSoilTypeComboBox.Items.Add(EpSoilsTable.Rows(i).Item(0))
        Next

        CheckStabilityButton.Enabled = False

        EraseResults()
        Refresh()
```

```
    End Sub

    Private Sub ProfileTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles ProfileTab.Enter
        HeightEntryBox.Text = HeightOfAnalysis
        HeightEntryBox.Select()

        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub EditSoilsButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles EditSoilsButton.Click
        SoilTypeEditMode = True
        SoilNumber = EpSoilsTable.Rows.Count
        For i = 0 To SoilNumber - 1
            If SoilsDataGridView.Rows(i).Selected Then SoilNumber = i
        Next

        If EpSoilsTable.Rows.Count = 0 Then
            MsgBox("There is no soil type to be edited.")
            SoilTypeEditMode = False
            Exit Sub
        ElseIf SoilNumber = EpSoilsTable.Rows.Count Then
            MsgBox("Please select type of soil to be edited.")
            SoilTypeEditMode = False
            Exit Sub
        End If


        My.Forms.SoilDataInput.ShowDialog()

        If SoilDataInput.DialogResult = DialogResult.OK Then
            SoilDisplayTable.Rows(SoilNumber).Item(0) = NewSoilLayerName
            SoilsDataGridView.ClearSelection()

            SoilTypeColumn.Items(SoilNumber) = NewSoilLayerName
            RetainingWallParameters.PassiveSoilTypeComboBox.Items(SoilNumber) =
NewSoilLayerName

        End If

        EraseResults()
        SoilTypeEditMode = False
    End Sub

    Private Sub Soils_Enter(ByVal sender As Object, ByVal e As System.EventArgs) Handles
Soils.Enter
        SoilsDataGridView.DataSource = SoilDisplayTable

        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub ExitToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ExitToolStripMenuItem.Click
```

```
        Close()
    End Sub

    Private Sub TheoryToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TheoryToolStripMenuItem.Click
        InputTabs.SelectedTab = TheoryTab
    End Sub

    Private Sub GrountToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles GrountToolStripMenuItem.Click
        InputTabs.SelectedTab = GroundTab
    End Sub

    Private Sub GeometryToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles GeometryToolStripMenuItem.Click
        InputTabs.SelectedTab = GeometyTab
    End Sub

    Private Sub ProfileToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ProfileToolStripMenuItem.Click
        InputTabs.SelectedTab = ProfileTab
    End Sub

    Private Sub SoilsToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles SoilsToolStripMenuItem.Click
        InputTabs.SelectedTab = Soils
    End Sub

    Private Sub AssignToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles AssignToolStripMenuItem.Click
        InputTabs.SelectedTab = AssignTab
    End Sub

    Private Sub GroundWaterToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles GroundWaterToolStripMenuItem.Click
        InputTabs.SelectedTab = GroundWater
    End Sub

    Private Sub SurchargeToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles SurchargeToolStripMenuItem.Click
        InputTabs.SelectedTab = SurchargeTab
    End Sub

    Private Sub RunAnalysisToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RunAnalysisToolStripMenuItem.Click
        RunAnalysisButton.PerformClick()
    End Sub

    Private Sub EditParametersToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles EditParametersToolStripMenuItem.Click
        InputTabs.SelectedTab = RetWallTab
    End Sub

    Private Sub AboutTheProgramToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles AboutTheProgramToolStripMenuItem.Click
        My.Forms.AboutForm.Show()
    End Sub
```

```
    Private Sub SoilToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles SoilToolStripMenuItem.Click
        InputTabs.SelectedTab = AnalysisTab
    End Sub

    Private Sub EraseResults()

        'Erasing Earth Pressure Parameters
        AnalysisCompleted = False
        CalculationsDataTable.Rows.Clear()
        CalculationParametersTable.Rows.Clear()


        'Erasing Retaining Wall Parameters
        RetainingWallCalculated = False

        ActivePressureTable.Rows.Clear()
        PassivePressureTable.Rows.Clear()
        WallPressureParametersTable.Rows.Clear()

        'ResultForcesTable.Rows.Clear()
        'SafetyFactorTable.Rows.Clear()
        StabilityParametersTable.Rows.Clear()

        'Recreating the Rows of StabiltyAnalsis table and Forces Data Table
        StabilityReportButton.Enabled = False
        OutputGridView.Visible = False
    End Sub

    Private Sub TheoryTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles TheoryTab.Enter
        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub AnalysisTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles AnalysisTab.Enter
        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub CreateNewAssignSoilsGridView()
        EraseResults()

        AssignDataGridView.Rows.Clear()
        AssignedSoilType.Rows.Clear()
        For i = 0 To GivenInterfaceTable.Rows.Count - 2
            AssignDataGridView.Rows.Add()
            AssignedSoilType.Rows.Add()
        Next
        RefreshAssignTexts()
        CheckStabilityButton.Enabled = False

    End Sub

    Private Sub RefreshAssignTexts()
```

```
        For i = 0 To GivenInterfaceTable.Rows.Count - 2
            AssignDataGridView.Rows(i).Cells(0).Value = GivenInterfaceTable.Rows(i +
1).Item(0) _
                                                                    -
GivenInterfaceTable.Rows(i).Item(0)

            AssignDataGridView.Rows(i).Cells(1).Value =
GivenInterfaceTable.Rows(i).Item(0) & " m - " & _
                                                       GivenInterfaceTable.Rows(i +
1).Item(0) & " m"
        Next
    End Sub

    Private Sub AssignDataGridView_CurrentCellDirtyStateChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles AssignDataGridView.CurrentCellDirtyStateChanged
        If AssignDataGridView.IsCurrentCellDirty Then
            EraseResults()
            AssignDataGridView.CommitEdit(DataGridViewDataErrorContexts.Commit)
            ReadAssignedSoils()
        End If
    End Sub

    Private Sub ReadAssignedSoils()
        Dim i, j As Integer
        'Converting the combobox selection into selected indexes
        For i = 0 To AssignDataGridView.Rows.Count - 1
            For j = 0 To EpSoilsTable.Rows.Count - 1
                If AssignDataGridView.Rows(i).Cells(2).Value =
EpSoilsTable.Rows(j).Item("Type of Soil") Then
                    AssignedSoilType.Rows(i).Item(0) = j
                End If
            Next
        Next
    End Sub

    Private Sub AssignTab_Enter(ByVal sender As Object, ByVal e As System.EventArgs)
Handles AssignTab.Enter
        RetainingWallDiagram = False
        If AnalysisCompleted = False Then OutputGridView.Visible = False
        Refresh()
    End Sub

    Private Sub GravityWallRadioButton_CheckedChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles GravityWallRadioButton.CheckedChanged
        EraseResults()
        Refresh()
    End Sub

    Private Sub ArrangeSoilPropertiesForRetWall()
        Dim i As Integer

        ActiveSoilsTable = EpSoilsTable.Copy
        If Theory = "Coulomb" Then
            TotalStressExists = False
            For i = 0 To AssignmentTable.Rows.Count - 1
                If ActiveSoilsTable.Rows(AssignmentTable.Rows(i).Item(0)).Item("Stress
State") = "Total Stress" Then TotalStressExists = True
            Next
```

```
            For i = 0 To ActiveSoilsTable.Rows.Count - 1
                Dim SqrtVariable, phi, delta, beta As Single
                phi = Math.PI * (ActiveSoilsTable.Rows(i).Item("Angle of Internal
Friction")) / 180
                delta = Math.PI * (ActiveSoilsTable.Rows(i).Item("Angle of Soil Structure
Friction")) / 180
                beta = BackfillInclinationBeta

                If (b3 - 0.15) > 0.00001 Or TotalStressExists Then
                    alpha = Math.PI / 2
                Else
                    alpha = Atan(1 / S2)
                End If

                SqrtVariable = Sqrt(Sin(phi + delta) * Sin(phi - beta) / Sin(alpha -
delta) / Sin(alpha + beta))
                ActiveSoilsTable.Rows(i).Item("ka") = (Sin(phi + alpha)) ^ 2 /
(Sin(alpha)) ^ 2 / Sin(alpha - delta) _
                    / (1 + SqrtVariable) ^ 2

                SqrtVariable = Sqrt(Sin(phi + delta) * Sin(phi + beta) / Sin(alpha +
delta) / Sin(alpha + beta))
                ActiveSoilsTable.Rows(i).Item("kp") = (Sin(alpha - phi)) ^ 2 /
(Sin(alpha)) ^ 2 / Sin(alpha + delta) _
                    / (1 - SqrtVariable) ^ 2
            Next
        End If



        PassiveSoilTable = EpSoilsTable.Copy
        If Theory = "Coulomb" Then
            TotalStressExists = False
            For i = 0 To AssignmentTable.Rows.Count - 1
                If PassiveSoilTable.Rows(AssignmentTable.Rows(i).Item(0)).Item("Stress
State") = "Total Stress" Then TotalStressExists = True
            Next

            For i = 0 To PassiveSoilTable.Rows.Count - 1
                Dim SqrtVariable, phi, delta, beta As Single
                phi = Math.PI * (PassiveSoilTable.Rows(i).Item("Angle of Internal
Friction")) / 180
                delta = Math.PI * (PassiveSoilTable.Rows(i).Item("Angle of Soil Structure
Friction")) / 180
                beta = BackfillInclinationBeta

                If (b1 - 0.15) > 0.00001 Or TotalStressExists Then
                    alpha = Math.PI / 2
                Else
                    alpha = Atan(1 / S1)
                End If

                SqrtVariable = Sqrt(Sin(phi + delta) * Sin(phi - beta) / Sin(alpha -
delta) / Sin(alpha + beta))
                PassiveSoilTable.Rows(i).Item("ka") = (Sin(phi + alpha)) ^ 2 /
(Sin(alpha)) ^ 2 / Sin(alpha - delta) _
                    / (1 + SqrtVariable) ^ 2
```

```
                SqrtVariable = Sqrt(Sin(phi + delta) * Sin(phi + beta) / Sin(alpha +
delta) / Sin(alpha + beta))
                PassiveSoilTable.Rows(i).Item("kp") = (Sin(alpha - phi)) ^ 2 /
(Sin(alpha)) ^ 2 / Sin(alpha + delta) _
                    / (1 - SqrtVariable) ^ 2
            Next

        End If

    End Sub

    Private Sub TensionCrackCheckBox_CheckedChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TensionCrackCheckBox.CheckedChanged
        If TensionCrackCheckBox.Checked Then
            HydrostaticDueToCrackEnabled = True
        Else
            HydrostaticDueToCrackEnabled = False
        End If
    End Sub

    Private Sub CalculateBearingCapacity()

        Dim CTerm, QTerm, BTerm As Single

        'Calculation of Bearing Capacity Factors
        Dim Nc, Ngamma, Nq As Single
        Nq = (Tan(Math.PI / 4 + FoundationPhi / 2)) ^ 2 * (Math.E) ^ (Math.PI *
Tan(FoundationPhi))
        If FoundationPhi = 0 Then
            Nc = 5.14
        Else
            Nc = (Nq - 1) / Tan(FoundationPhi)
        End If
        Ngamma = 2 * (Nq + 1) * Tan(FoundationPhi)

        'Calculation of depth factors
        Dim Fcd, Fqd, FgammaD As Single
        Dim Df As Single = PassiveSoilThickness
        FgammaD = 1

        If Df <= b Then
            If FoundationPhi = 0 Then
                Fqd = 1
                Fcd = 1 + 0.4 * Df / b
            Else
                Fqd = 1 + 2 * Tan(FoundationPhi) * (1 - Sin(FoundationPhi)) ^ 2 * Df / b
                Fcd = Fqd - (1 - Fqd) / Nc / Tan(FoundationPhi)
            End If
        Else
            If FoundationPhi = 0 Then
                Fqd = 1
                Fcd = 1 + 0.4 * Atan(Df / b)
            Else
                Fqd = 1 + 2 * Tan(FoundationPhi) * (1 - Sin(FoundationPhi)) ^ 2 * Atan(Df
/ b)
                Fcd = Fqd - (1 - Fqd) / Nc / Tan(FoundationPhi)
            End If
```

```
        End If

        'Calculation of Inclination Factors
        Dim Fci, Fqi, FgammaI As Single
        Dim HorizontalLoad, VerticalLoad, LoadInclination As Single
        VerticalLoad = VerticalReaction
        HorizontalLoad = ResultForcesTable.Rows(3).Item(1)
        LoadInclination = Atan(HorizontalLoad / VerticalLoad)

        Fci = (1 - 2 * LoadInclination / Math.PI) ^ 2
        Fqi = Fci
        If FoundationPhi > LoadInclination And FoundationPhi > 0 Then
            FgammaI = (1 - LoadInclination / FoundationPhi)
        Else
            FgammaI = 1
        End If

        'Calculation of q and Gamma (with consideration of GWT depth)
        Dim GammaPassive, GammaSaturatedPassive As Single
        GammaPassive = PassivePressureTable.Rows(0).Item("Bulk Unit Weight")
        GammaSaturatedPassive = PassivePressureTable.Rows(0).Item("Saturated Unit Weight")

        Dim q, Gamma As Single

        If GroundWaterDepthFront >= PassiveSoilThickness + b Then
            q = PassiveSoilThickness * GammaPassive + PassiveSurcharge
            Gamma = FoundationGammaBulk
        ElseIf GroundWaterDepthFront > PassiveSoilThickness Then
            q = PassiveSoilThickness * GammaPassive + PassiveSurcharge
            Gamma = (FoundationGammaSaturated - UnitWeightOfWater) _
                + (GroundWaterDepthFront - PassiveSoilThickness) / b *
(FoundationGammaBulk - (FoundationGammaSaturated - UnitWeightOfWater))
        Else
            q = GroundWaterDepthFront * GammaPassive _
                + (PassiveSoilThickness - GroundWaterDepthFront) * (GammaSaturatedPassive
- UnitWeightOfWater) _
                + PassiveSurcharge
            Gamma = FoundationGammaSaturated - UnitWeightOfWater
        End If

        CTerm = FoundationCohesion * Nc * 1 * Fcd * Fci
        QTerm = q * Nq * 1 * Fqd * Fqi
        BTerm = 0.5 * Gamma * b * Ngamma * 1 * FgammaD * FgammaI

        UltimateBearingCapacity = CTerm + QTerm + BTerm

        Dim RowNumber As UInteger
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber = StabilityParametersTable.Rows.Count - 1
        StabilityParametersTable.Rows(RowNumber).Item(0) = "Calculation of Allowable
Bearing Capacity"

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        RowNumber += 1
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Cohesion of Foundation
Soil"
```

```
StabilityParametersTable.Rows(RowNumber + 1).Item(1) = "kPa"
StabilityParametersTable.Rows(RowNumber + 1).Item(2) = FoundationCohesion

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Nc"
StabilityParametersTable.Rows(RowNumber + 2).Item(2) = Nc

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 3).Item(0) = "Fcs"
StabilityParametersTable.Rows(RowNumber + 3).Item(2) = 1

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 4).Item(0) = "Fcd"
StabilityParametersTable.Rows(RowNumber + 4).Item(2) = Fcd

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 5).Item(0) = "Fci"
StabilityParametersTable.Rows(RowNumber + 5).Item(2) = Fci

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 6).Item(0) = "C-term"
StabilityParametersTable.Rows(RowNumber + 6).Item(1) = "kPa"
StabilityParametersTable.Rows(RowNumber + 6).Item(2) = CTerm

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows.Add()
RowNumber = StabilityParametersTable.Rows.Count - 1
StabilityParametersTable.Rows(RowNumber).Item(0) = "q"
StabilityParametersTable.Rows(RowNumber).Item(1) = "kPa"
StabilityParametersTable.Rows(RowNumber).Item(2) = q

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "Nq"
StabilityParametersTable.Rows(RowNumber + 1).Item(2) = Nq

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Fqs"
StabilityParametersTable.Rows(RowNumber + 2).Item(2) = 1

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 3).Item(0) = "Fqd"
StabilityParametersTable.Rows(RowNumber + 3).Item(2) = Fqd

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 4).Item(0) = "Fqi"
StabilityParametersTable.Rows(RowNumber + 4).Item(2) = Fqi

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows(RowNumber + 5).Item(0) = "q-term"
StabilityParametersTable.Rows(RowNumber + 5).Item(1) = "kPa"
StabilityParametersTable.Rows(RowNumber + 5).Item(2) = QTerm

StabilityParametersTable.Rows.Add()
StabilityParametersTable.Rows.Add()
RowNumber = StabilityParametersTable.Rows.Count - 1
StabilityParametersTable.Rows(RowNumber).Item(0) = "Unit Weight"
StabilityParametersTable.Rows(RowNumber).Item(1) = "kN/m3"
StabilityParametersTable.Rows(RowNumber).Item(2) = Gamma
```

```
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 1).Item(0) = "B"
        StabilityParametersTable.Rows(RowNumber + 1).Item(2) = b

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 2).Item(0) = "Nγ"
        StabilityParametersTable.Rows(RowNumber + 2).Item(2) = Ngamma
        'gamma"
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 3).Item(0) = "Fγs"
        StabilityParametersTable.Rows(RowNumber + 3).Item(2) = 1

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 4).Item(0) = "Fγd"
        StabilityParametersTable.Rows(RowNumber + 4).Item(2) = FgammaD

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 5).Item(0) = "Fγi"
        StabilityParametersTable.Rows(RowNumber + 5).Item(2) = FgammaI

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 6).Item(0) = "B-term"
        StabilityParametersTable.Rows(RowNumber + 6).Item(1) = "kPa"
        StabilityParametersTable.Rows(RowNumber + 6).Item(2) = BTerm

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 8).Item(0) = "Ultimate Bearing
Capacity"
        StabilityParametersTable.Rows(RowNumber + 8).Item(1) = "kPa"
        StabilityParametersTable.Rows(RowNumber + 8).Item(2) = UltimateBearingCapacity

        StabilityParametersTable.Rows.Add()
        StabilityParametersTable.Rows(RowNumber + 9).Item(0) = "Factor of Safety for
Bearing capacity"
        StabilityParametersTable.Rows(RowNumber + 9).Item(2) = BearingCapacityFS

    End Sub
End Class
```

Public Variable Module

```
Module PublicVariablesModule
    'Public NumberOfInterfaces As Integer = 2
    'Public SoilTypeNumber As Integer
    Public InterfaceMatix As Single(,)

    Public NewInterfaceDepth As Single
    Public NewSoilLayerName As String

    Public HeightOfAnalysis As Single
    Public UnitWeightOfWater As Single

    Public CalculationsDataTable As New DataTable("CalculationsTable")
    'Public SurchargeTable As New DataTable
    Public CalculateSoilsPropertiesTable As New DataTable

    Public EpSoilsTable, ActiveSoilsTable, PassiveSoilTable As New DataTable
    'Public ActiveSoilsTable, PassiveSoilTable As New DataTable

    Public SoilDisplayTable As New DataTable
    'Public StabilityParameterTable As New DataTable

    Public RefinedSurchargeGraphics As Boolean = False

    Public HorizontalResultantForce, HorizontalResultantLocation As Single
    Public VerticalResultantForce, VerticalResultantLocation As Single

    Public BackfillInclinationBeta, WallInclinationAlpha As Single
    Public AnalysisType As String
    Public Theory As String = ""

    'Public ProfileWidth As Single
    Public SoilStructureFrictionAngle As Single
    Public alpha As Single

    Public EarthPressureTable As New DataTable("EarthPressureTable")
    Public ActivePressureTable As New DataTable("ActivePressureTable")
    Public PassivePressureTable As New DataTable("PassivePressureTable")

    Public RetainingWallHeight As Single
    Public GammaRtWall As Single
    Public PassiveSoilType As UInteger
    Public PassiveSoilThickness As Single
    Public PassiveSurcharge As Single
    Public AllowableBearingCapacity, UltimateBearingCapacity As Single
    Public FoundationFriction As Single
    Public FoundationAdhesion As Single
    Public AllowedSlipFS, AllowedOverturnFS, BearingCapacityFS As Single
    Public ResultForcesTable As New DataTable
    Public SafetyFactorTable As New DataTable

    Public RetainingWallDiagram As Boolean = False

    Public H, d, S1, S2, b, b1, b2, b3 As Single

    Public CalculationParametersTable As New DataTable
    'Public EarthPressureParametersTable As New DataTable
```

```
    Public WallPressureParametersTable As New DataTable
    Public StabilityParametersTable As New DataTable

    Public SoilTypeEditMode As Boolean = False
    Public SoilNumber As Integer

    Public AssignedSoilType As New DataTable

    Public PassiveAssigned As Boolean = False

    Public ActiveSoilDisplayTable As New DataTable
    Public PassiveSoilDisplayTable As New DataTable

    Public BearingCapacityByUser As Boolean
    Public GroundWaterDepthFront As Single
    Public FoundationCohesion, FoundationPhi, FoundationGammaBulk,
FoundationGammaSaturated As Single

End Module
```

Retaining Wall Parameters Form

```
Imports System.Math

Public Class RetainingWallParameters
    Dim TempoAllowableBearingCapacityByUser As Boolean

    Private Sub OkButton_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles OkButton.Click
        Try
            RetainingWallHeight = MainForm.HeightsGridView.Rows(0).Cells(0).Value

            If GammaRtWallTextBox.Text < 0 Then
                MsgBox("Unti Weight of retaining wall cannot be negative.")
                Exit Sub
            End If
            GammaRtWall = GammaRtWallTextBox.Text

            PassiveSoilType = PassiveSoilTypeComboBox.SelectedIndex

            If PassiveSoilThicknessTextBox.Text < 0 Then
                MsgBox("Thickness of soil on the front face cannot be negative.")
                Exit Sub
            ElseIf PassiveSoilThicknessTextBox.Text > H Then
                MsgBox("Thickness of soil on the front face cannot be larger than the
Height of Wall.")
                Exit Sub
            End If
            PassiveSoilThickness = PassiveSoilThicknessTextBox.Text

            If PassiveSurchargeTextBox.Text = "" Then
                PassiveSurcharge = 0
            ElseIf PassiveSurchargeTextBox.Text < 0 Then
                MsgBox("Magnitude of surcharge load cannot be negative.")
                Exit Sub
            End If
            PassiveSurcharge = PassiveSurchargeTextBox.Text

            If GwtFront.Text = "" Then
                GroundWaterDepthFront = 0
            ElseIf GwtFront.Text < 0 Then
                MsgBox("Depth of Ground Water cannot be negative.")
                Exit Sub
            End If
            GroundWaterDepthFront = GwtFront.Text


            If FoundationFrictionTextBox.Text < 0 Then
                MsgBox("Angle of Soil-Structure friction cannot be negative.")
                Exit Sub
            End If
            FoundationFriction = FoundationFrictionTextBox.Text * Math.PI / 180

            If AdhesionTextBox.Text < 0 Then
                MsgBox("Foundation Adhesion cannot be negative.")
                Exit Sub
```

```
            End If
            FoundationAdhesion = AdhesionTextBox.Text


            If BearingCapacityByUser Then
                If QUltTextBox.Text < 0 Then
                    MsgBox("Ultimate Bearing Capacity cannot be negative.")
                    Exit Sub
                End If
                UltimateBearingCapacity = QUltTextBox.Text
            Else
                If CohesionTextBox.Text < 0 Then
                    MsgBox("The value of Cohesion cannot be negative.")
                    Exit Sub
                ElseIf CohesionTextBox.Text < AdhesionTextBox.Text Then
                    MsgBox("Cohesion cannot be less than Adhesion.")
                    Exit Sub
                End If
                FoundationCohesion = CohesionTextBox.Text

                If InternalFrictionTextBox.Text < 0 Then
                    MsgBox("The value of Internal Friction Angle cannot be negative.")
                    Exit Sub
                ElseIf InternalFrictionTextBox.Text < FoundationFrictionTextBox.Text Then
                    MsgBox("Angle of Internal Friction cannot be less than Soil-Structure
Friction Angle.")
                    Exit Sub
                End If
                FoundationPhi = InternalFrictionTextBox.Text * Math.PI / 180

                If BulkGammaTextBox.Text < 0 Then
                    MsgBox("Bulk Unit Weight cannot be negative.")
                    Exit Sub
                End If
                FoundationGammaBulk = BulkGammaTextBox.Text

                If SaturatedGammaTextBox.Text = "" Then
                    SaturatedGammaTextBox.Text = BulkGammaTextBox.Text
                ElseIf SaturatedGammaTextBox.Text < FoundationGammaBulk Then
                    MsgBox("Saturated Unit Weight cannot be less than Bulk Unit Weight.")
                    Exit Sub
                End If
            FoundationGammaSaturated = SaturatedGammaTextBox.Text
            End If

            If SlipSafetyFactorTextBox.Text < 0 Then
                MsgBox("Factor of Safety against Slip cannot be negative.")
                Exit Sub
            End If
            AllowedSlipFS = SlipSafetyFactorTextBox.Text

            If OverturningSafetyFactorTextBox.Text < 0 Then
                MsgBox("Factor of Safety against Overturning cannot be negative.")
                Exit Sub
            End If
            AllowedOverturnFS = OverturningSafetyFactorTextBox.Text

            If BearingSafetyTextBox.Text < 0 Then
```

```
                MsgBox("Factor of Safety for Bearing Capacity cannot be negative.")
                Exit Sub
            End If
            BearingCapacityFS = BearingSafetyTextBox.Text

            DialogResult = DialogResult.OK
        Catch ex As Exception
            MsgBox("Parameters are either incomplete or contain an invalid value!")
        End Try

    End Sub

    Private Sub CancelBttn_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CancelBttn.Click
        If TempoAllowableBearingCapacityByUser Then
            BCknownRadioButton.Checked = True
        Else
            MeyerhofRadioButton.Checked = True
        End If
        DialogResult = DialogResult.Cancel
    End Sub

    Private Sub BCknownRadioButton_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BCknownRadioButton.CheckedChanged
        If BCknownRadioButton.Checked Then

            BearingCapacityByUser = True

            Label18.Visible = True
            Label19.Visible = True
            Label20.Visible = True
            QUltTextBox.Visible = True
            Label21.Visible = True

            Label10.Visible = False
            CohesionTextBox.Visible = False
            Label11.Visible = False

            Label24.Visible = False
            InternalFrictionTextBox.Visible = False
            Label22.Visible = False

            Label25.Visible = False
            BulkGammaTextBox.Visible = False
            Label27.Visible = False
            Label26.Visible = False

            Label28.Visible = False
            SaturatedGammaTextBox.Visible = False
            Label30.Visible = False
            Label29.Visible = False

        Else

            BearingCapacityByUser = False

            Label18.Visible = False
            Label19.Visible = False
```

```
            Label20.Visible = False
            QUltTextBox.Visible = False
            Label21.Visible = False

            Label10.Visible = True
            CohesionTextBox.Visible = True
            Label11.Visible = True

            Label24.Visible = True
            InternalFrictionTextBox.Visible = True
            Label22.Visible = True

            Label25.Visible = True
            BulkGammaTextBox.Visible = True
            Label27.Visible = True
            Label26.Visible = True

            Label28.Visible = True
            SaturatedGammaTextBox.Visible = True
            Label30.Visible = True
            Label29.Visible = True

        End If
    End Sub


    Private Sub RetainingWallParameters_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        TempoAllowableBearingCapacityByUser = BearingCapacityByUser
    End Sub

    Private Sub Label12_Click(sender As Object, e As EventArgs) Handles Label12.Click

    End Sub
End Class
```

Soil Data Input Form

```
Imports System.Math

Public Class SoilDataInput

    Private Sub SoilInputOkButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles SoilInputOkButton.Click
        Dim i As UInteger

        Try
            i = 0
            For Each SoilTableRow As DataRow In EpSoilsTable.Rows
                If SoilTableRow.Item("Type of Soil") = SoilNameTextBox.Text And
SoilNumber <> i Then
                    MsgBox("Name of Soil has already been used.")
                    Exit Sub
                End If
                i += 1
            Next SoilTableRow
            If SoilNameTextBox.Text = "" Then
                MsgBox("Name of Soil has not been provided.")
                Exit Sub
            End If

            EpSoilsTable.Rows.Add()
            i = EpSoilsTable.Rows.Count - 1
            EpSoilsTable.Rows(i).Item("Type of Soil") = SoilNameTextBox.Text


            EpSoilsTable.Rows(i).Item("Bulk Unit Weight") = UnitWtTextBox.Text

            If SaturatedUnitWeightTextBox.Text = "" Then
                EpSoilsTable.Rows(i).Item("Saturated Unit Weight") = UnitWtTextBox.Text
            ElseIf SaturatedUnitWeightTextBox.Text < UnitWtTextBox.Text Then
                MsgBox("Satrurated Unit Weight cannot be less than Bulk Unit Weight")
                EpSoilsTable.Rows.RemoveAt(i)
                Exit Sub
            Else
                EpSoilsTable.Rows(i).Item("Saturated Unit Weight") =
SaturatedUnitWeightTextBox.Text
            End If

            EpSoilsTable.Rows(i).Item("Stress State") = StressStateComboBox.SelectedItem

            EpSoilsTable.Rows(i).Item("Angle of Backfill Soil") = BackfillInclinationBeta
* 180 / Math.PI

            If BackfillInclinationBeta > Math.PI * (InternalFrictionTextBox.Text) / 180
Then
                MsgBox("Inclination of backfill has to be less than than Phi.")
                EpSoilsTable.Rows.RemoveAt(i)
                Exit Sub
            End If

            If InternalFrictionTextBox.Text < 0 Then
```

```
                MsgBox("Angle of Internal Friction cannot be negative!")
                EpSoilsTable.Rows.RemoveAt(i)
                Exit Sub
            End If

            If InternalFrictionTextBox.Text > 80 Then
                MsgBox("The maximum allowed value of Phi is 80 degrees!")
                EpSoilsTable.Rows.RemoveAt(i)
                Exit Sub
            End If
            EpSoilsTable.Rows(i).Item("Angle of Internal Friction") =
InternalFrictionTextBox.Text

            If CohesionTextBox.Text < 0 Then
                MsgBox("Cohesion cannot be negative!")
                EpSoilsTable.Rows.RemoveAt(i)
                Exit Sub
            End If
            EpSoilsTable.Rows(i).Item("Cohesion") = CohesionTextBox.Text


            If Theory = "Coulomb" Or Theory = "KeriselAbsi" Then
                If AdhesionBox.Text < 0 Then
                    MsgBox("Soil-Structure Adhesion cannot be negative!")
                    EpSoilsTable.Rows.RemoveAt(i)
                    Exit Sub
                End If
                EpSoilsTable.Rows(i).Item("Adhesion") = AdhesionBox.Text

                If StructureFrictionBox.Text < 0 Then
                    MsgBox("Angle of Soil-Structure Friction cannot be negative")
                    EpSoilsTable.Rows.RemoveAt(i)
                    Exit Sub
                End If
                EpSoilsTable.Rows(i).Item("Angle of Soil Structure Friction") =
StructureFrictionBox.Text
            End If


            If AnalysisType = "At Rest Earth Pressure" Then
                EpSoilsTable.Rows(i).Item("Theory") = "At Rest Pressure"
                EpSoilsTable.Rows(i).Item("Soil AtRest Condition") =
SoilTypeAtRestComboBox.SelectedItem
                Select Case SoilTypeAtRestComboBox.SelectedIndex
                    Case 0  'cohesionless
                        EpSoilsTable.Rows(i).Item("Ko") = (1 - Sin(Math.PI *
(InternalFrictionTextBox.Text) / 180)) _
                                                              / (1 +
Sin(BackfillInclinationBeta))
                    Case 1  'cohesive
                        Dim PoissonsRatio As Single = AtRestPropertyTextBox.Text
                        If PoissonsRatio >= 0.5 Or PoissonsRatio < 0 Then
                            MsgBox("The Value of Poisson's Ratio is Invalid!")
                            EpSoilsTable.Rows.RemoveAt(i)
                            Exit Sub
                        End If
                        EpSoilsTable.Rows(i).Item("Poissons Ratio") = PoissonsRatio
```

```
                        EpSoilsTable.Rows(i).Item("Ko") = PoissonsRatio / (1 -
PoissonsRatio)
                    Case 2  'normally consolidated
                        Dim PlasticityIndex As Single = AtRestPropertyTextBox.Text
                        EpSoilsTable.Rows(i).Item("PI") = PlasticityIndex
                        If 0 <= PlasticityIndex And PlasticityIndex < 40 Then
                            EpSoilsTable.Rows(i).Item("Ko") = 0.4 + 0.007 *
PlasticityIndex
                        ElseIf 40 <= PlasticityIndex Then
                            EpSoilsTable.Rows(i).Item("Ko") = 0.64 + 0.001 *
PlasticityIndex
                        Else
                            MsgBox("The Value of PI is Invalid!")
                            EpSoilsTable.Rows.RemoveAt(i)
                            Exit Sub
                        End If
                    Case 3  'Overconsolidated
                        If AtRestPropertyTextBox.Text < 0 Then
                            MsgBox("Overconsolidation Ratio cannot be negative!")
                            EpSoilsTable.Rows.RemoveAt(i)
                            Exit Sub
                        End If
                        EpSoilsTable.Rows(i).Item("OCR") = AtRestPropertyTextBox.Text
                        EpSoilsTable.Rows(i).Item("Ko") = (1 -
Sin(InternalFrictionTextBox.Text * Math.PI / 180)) * _
                                                        AtRestPropertyTextBox.Text ^
Sin(InternalFrictionTextBox.Text * Math.PI / 180)
                    Case 4  'Ko value given
                        EpSoilsTable.Rows(i).Item("Ko") = AtRestPropertyTextBox.Text
                End Select
            Else
                Select Case Theory
                    Case "Rankine"
                        Dim SqrtVariable As Single
                        SqrtVariable = Sqrt((Cos(BackfillInclinationBeta) ^ 2) -
(Cos(Math.PI * InternalFrictionTextBox.Text / 180) ^ 2))
                        EpSoilsTable.Rows(i).Item("ka") = Cos(BackfillInclinationBeta) _
                            * (Cos(BackfillInclinationBeta) - SqrtVariable) _
                            / (Cos(BackfillInclinationBeta) + SqrtVariable)

                        EpSoilsTable.Rows(i).Item("kp") = Cos(BackfillInclinationBeta) _
                            * (Cos(BackfillInclinationBeta) + SqrtVariable) _
                            / (Cos(BackfillInclinationBeta) - SqrtVariable)
                        EpSoilsTable.Rows(i).Item("Theory") = "Rankine"
                    Case "Bell"
                        Dim SqrtVariable As Single
                        SqrtVariable = Sqrt((Cos(BackfillInclinationBeta) ^ 2) -
(Cos(Math.PI * InternalFrictionTextBox.Text / 180) ^ 2))
                        EpSoilsTable.Rows(i).Item("ka") = Cos(BackfillInclinationBeta) _
                            * (Cos(BackfillInclinationBeta) - SqrtVariable) _
                            / (Cos(BackfillInclinationBeta) + SqrtVariable)

                        EpSoilsTable.Rows(i).Item("kp") = Cos(BackfillInclinationBeta) _
                            * (Cos(BackfillInclinationBeta) + SqrtVariable) _
                            / (Cos(BackfillInclinationBeta) - SqrtVariable)
                        EpSoilsTable.Rows(i).Item("Theory") = "Bell"
                    Case "Coulomb"
                        Dim SqrtVariable, phi, delta, beta As Single
```

```
                        phi = Math.PI * (InternalFrictionTextBox.Text) / 180
                        delta = Math.PI * (StructureFrictionBox.Text) / 180
                        alpha = Math.PI / 2 - WallInclinationAlpha
                        beta = BackfillInclinationBeta

                        SqrtVariable = Sqrt(Sin(phi + delta) * Sin(phi - beta) /
Sin(alpha - delta) / Sin(alpha + beta))
                        EpSoilsTable.Rows(i).Item("ka") = (Sin(phi + alpha)) ^ 2 /
(Sin(alpha)) ^ 2 / Sin(alpha - delta) _
                            / (1 + SqrtVariable) ^ 2

                        SqrtVariable = Sqrt(Sin(phi + delta) * Sin(phi + beta) /
Sin(alpha + delta) / Sin(alpha + beta))
                        EpSoilsTable.Rows(i).Item("kp") = (Sin(alpha - phi)) ^ 2 /
(Sin(alpha)) ^ 2 / Sin(alpha + delta) _
                            / (1 - SqrtVariable) ^ 2

                        EpSoilsTable.Rows(i).Item("Theory") = "Coulomb"
                    Case "KeriselAbsi"
                        If AtRestPropertyTextBox.Text < 0 Then
                            MsgBox("kp cannot be negative!")
                            EpSoilsTable.Rows.RemoveAt(i)
                            Exit Sub
                        Else
                            EpSoilsTable.Rows(i).Item("kp") = AtRestPropertyTextBox.Text
                        End If

                        If AtRestBox2.Text < 0 Then
                            MsgBox("ka cannot be negative!")
                            EpSoilsTable.Rows.RemoveAt(i)
                            Exit Sub
                        Else
                            EpSoilsTable.Rows(i).Item("ka") = AtRestBox2.Text
                        End If
                        EpSoilsTable.Rows(i).Item("Theory") = "Kerisel and Absi"
                End Select
            End If

        Catch ex As Exception
            MsgBox("Error! Soil data input is either incomplete or contains an invalid
value.")
            EpSoilsTable.Rows.RemoveAt(i)
            Exit Sub
        End Try

        If SoilTypeEditMode = True Then
            Dim j As Integer
            For j = 0 To EpSoilsTable.Columns.Count - 1
                EpSoilsTable.Rows(SoilNumber).Item(j) = EpSoilsTable.Rows(i).Item(j)
            Next
            EpSoilsTable.Rows.RemoveAt(i)
        End If

        NewSoilLayerName = SoilNameTextBox.Text
        DialogResult = DialogResult.OK

    End Sub
```

```
    Private Sub SoilDataInput_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        StressStateComboBox.Enabled = True
        InternalFrictionTextBox.Enabled = True
        CohesionTextBox.Enabled = True
        StructureFrictionBox.Enabled = True
        AdhesionBox.Enabled = True

        If Theory = "KeriselAbsi" Then

            GroupBox3.Text = "Earth Pressure Coefficients (Kerisel and Absi)"

            Label12.Visible = False
            SoilTypeAtRestComboBox.Visible = False

            Label13.Text = "Passive Pressure Coeffcient"
            Label13.Top = 30
            Label13.Visible = True

            Label14.Text = "Kp="
            Label14.Top = 30
            Label14.Visible = True

            AtRestPropertyTextBox.Top = 30
            AtRestPropertyTextBox.Visible = True

            Label32.Text = "Active Pressure Coeffcient"
            Label32.Top = 60
            Label32.Visible = True

            Label31.Text = "Ka="
            Label31.Top = 60
            Label31.Visible = True

            AtRestBox2.Visible = True
            AtRestBox2.Top = 60

            ChartButton.Visible = True
        Else
            'theory<>Kerisel Absi
            GroupBox3.Text = "Parameters for Pressure At Rest"

            Label12.Visible = True
            SoilTypeAtRestComboBox.Visible = True
            SoilTypeAtRestComboBox.Enabled = True

            Label13.Top = 65
            Label13.Visible = True

            Label14.Top = 65
            Label14.Visible = True

            AtRestPropertyTextBox.Top = 65
            AtRestPropertyTextBox.Visible = True

            Label32.Visible = False
            Label31.Visible = False
            AtRestBox2.Visible = False
```

```
            ChartButton.Visible = False

            If AnalysisType <> "At Rest Earth Pressure" Then
                Label12.Enabled = False
                SoilTypeAtRestComboBox.Enabled = False

                Label13.Visible = False
                Label14.Visible = False
                AtRestPropertyTextBox.Visible = False

            Else
                Label12.Enabled = True
                SoilTypeAtRestComboBox.Enabled = True
                SelectSoilTypeAtRest()
            End If
        End If

        If SoilTypeEditMode Then
            Dim i As Integer
            i = SoilNumber

            SoilNameTextBox.Text = EpSoilsTable.Rows(i).Item("Type of Soil")
            UnitWtTextBox.Text = EpSoilsTable.Rows(i).Item("Bulk Unit Weight")
            SaturatedUnitWeightTextBox.Text = EpSoilsTable.Rows(i).Item("Saturated Unit
Weight")

            If EpSoilsTable.Rows(i).Item("Stress State") = "Total Stress" Then
                StressStateComboBox.SelectedIndex = 0
            Else
                StressStateComboBox.SelectedIndex = 1
            End If

            InternalFrictionTextBox.Text = EpSoilsTable.Rows(i).Item("Angle of Internal
Friction")
            CohesionTextBox.Text = EpSoilsTable.Rows(i).Item("Cohesion")

            Try
                AdhesionBox.Text = EpSoilsTable.Rows(i).Item("Adhesion")
            Catch ex As Exception
            End Try

            Try
                AdhesionBox.Text = EpSoilsTable.Rows(i).Item("Adhesion")
            Catch ex As Exception
            End Try

            If AnalysisType = "At Rest Earth Pressure" Then
                Select Case EpSoilsTable.Rows(i).Item("Soil AtRest Condition")
                    Case "Cohesionless"
                        SoilTypeAtRestComboBox.SelectedIndex = 0
                    Case "Cohesive"
                        SoilTypeAtRestComboBox.SelectedIndex = 1
                    Case "Normally Consolidated"
                        SoilTypeAtRestComboBox.SelectedIndex = 2
                    Case "Overconsolidated"
                        SoilTypeAtRestComboBox.SelectedIndex = 3
                    Case "Ko Value Known"
```

```
                    SoilTypeAtRestComboBox.SelectedIndex = 4
                End Select
            ElseIf Theory = "KeriselAbsi" Then
                AtRestPropertyTextBox.Text = EpSoilsTable.Rows(i).Item("kp")
                AtRestBox2.Text = EpSoilsTable.Rows(i).Item("ka")
            End If

            SaturatedUnitWeightTextBox.Text = EpSoilsTable.Rows(i).Item("Saturated Unit 
Weight")

            If UnitWeightOfWater = 10 Then
                WaterWeightComboBox.SelectedIndex = 1
            Else
                WaterWeightComboBox.SelectedIndex = 0
            End If
        End If

        ApplySoilParameterConstraints()
        SoilNameTextBox.Select()
    End Sub

    Private Sub SoilInputCancelButton_Click(ByVal sender As System.Object, ByVal e As 
System.EventArgs) Handles SoilInputCancelButton.Click
        DialogResult = DialogResult.Cancel
        Me.Close()
    End Sub

    Private Sub SoilTypeAtRestComboBox_SelectedIndexChanged(ByVal sender As System.Object, 
ByVal e As System.EventArgs) Handles SoilTypeAtRestComboBox.SelectedIndexChanged
        SelectSoilTypeAtRest()
    End Sub

    Private Sub WaterWeightComboBox_SelectedIndexChanged(ByVal sender As System.Object, 
ByVal e As System.EventArgs) Handles WaterWeightComboBox.SelectedIndexChanged
        Select Case WaterWeightComboBox.SelectedIndex
            Case 0
                UnitWeightOfWater = 9.81
            Case 1
                UnitWeightOfWater = 10
        End Select
    End Sub

    Private Sub UnitWtTextBox_TextChanged(ByVal sender As System.Object, ByVal e As 
System.EventArgs) Handles UnitWtTextBox.TextChanged
        If UnitWtTextBox.Text <> "" Then
            Dim test As Single
            Try
                test = 1 / Sqrt(UnitWtTextBox.Text)
            Catch ex As Exception
                MsgBox("Entry for Unit Weight is invalid!")
                UnitWtTextBox.Text = ""
            End Try
        End If
    End Sub

    Private Sub StressStateComboBox_SelectedIndexChanged(ByVal sender As System.Object, 
ByVal e As System.EventArgs) Handles StressStateComboBox.SelectedIndexChanged
        ApplySoilParameterConstraints()
```

```
    End Sub

    Private Sub ChartButton_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ChartButton.Click
        My.Forms.KeriselAbsiChart.Show()
    End Sub
    Private Sub ApplySoilParameterConstraints()
        Select Case StressStateComboBox.SelectedItem
            Case "Effective Stress"
                Label25.Text = "eff"
                Label26.Text = "ef"
                InternalFrictionTextBox.Enabled = True
                StructureFrictionBox.Enabled = True
            Case "Total Stress"
                Label25.Text = "U"
                Label26.Text = "U"
                InternalFrictionTextBox.Text = "0"
                StructureFrictionBox.Text = "0"
                InternalFrictionTextBox.Enabled = False
                StructureFrictionBox.Enabled = False
        End Select
        If AnalysisType = "At Rest Earth Pressure" Then
            StressStateComboBox.SelectedIndex = 1
            StressStateComboBox.Enabled = False

            If BackfillInclinationBeta > 0 Then
                CohesionTextBox.Text = 0
                CohesionTextBox.Enabled = False
            End If

            StructureFrictionBox.Text = ""
            StructureFrictionBox.Enabled = False

            AdhesionBox.Text = ""
            AdhesionBox.Enabled = False
        ElseIf Theory = "Rankine" Then
            StressStateComboBox.SelectedIndex = 1
            StressStateComboBox.Enabled = False

            CohesionTextBox.Text = 0
            CohesionTextBox.Enabled = False

            StructureFrictionBox.Text = ""
            StructureFrictionBox.Enabled = False

            AdhesionBox.Text = ""
            AdhesionBox.Enabled = False

            SoilTypeAtRestComboBox.SelectedIndex = 0
        ElseIf Theory = "Bell" Then
            'StressStateComboBox.SelectedIndex = 1
            'StressStateComboBox.Enabled = False

            StructureFrictionBox.Text = ""
            StructureFrictionBox.Enabled = False

            AdhesionBox.Text = ""
            AdhesionBox.Enabled = False
```

```
            'SoilTypeAtRestComboBox.SelectedIndex = 0
        ElseIf Theory = "Coulomb" Then
        ElseIf Theory = "KeriselAbsi" Then
            StressStateComboBox.SelectedIndex = 1
            StressStateComboBox.Enabled = False
        End If

        If BackfillInclinationBeta > 0 Then
            SoilTypeAtRestComboBox.SelectedIndex = 0
            SoilTypeAtRestComboBox.Enabled = False
        End If

        If StressStateComboBox.SelectedIndex = 0 Then

        End If
    End Sub

    Private Sub SelectSoilTypeAtRest()
        Select Case SoilTypeAtRestComboBox.SelectedIndex
            Case 0
                Label13.Visible = False
                Label14.Visible = False
                AtRestPropertyTextBox.Visible = False
            Case 1
                Label13.Text = "Poisson's ratio"
                Label13.Visible = True
                Label14.Text = "v="
                Label14.Visible = True
                AtRestPropertyTextBox.Visible = True
            Case 2
                Label13.Text = "Plasticity Index"
                Label13.Visible = True
                Label14.Text = "P.I.="
                Label14.Visible = True
                AtRestPropertyTextBox.Visible = True
            Case 3
                Label13.Text = "Overconsolidation ratio:"
                Label13.Visible = True
                Label14.Text = "OCR="
                Label14.Visible = True
                AtRestPropertyTextBox.Visible = True
            Case 4
                Label13.Text = "At rest Pressure Coefficient"
                Label13.Visible = True
                Label14.Text = "Ko="
                Label14.Visible = True
                AtRestPropertyTextBox.Visible = True
        End Select
    End Sub

End Class
```

Soil Removal Warning Form

```
Public Class SoilRemovedWarningForm
    Dim MyTimer As New System.Windows.Forms.Timer

    Private Sub SoilRemovedWarningForm_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        AddHandler MyTimer.Tick, AddressOf TimerEventProcessor
        MyTimer.Interval = 3500
        MyTimer.Enabled = True
        MyTimer.Start()
    End Sub
    Private Sub TimerEventProcessor()
        Me.Close()
    End Sub
End Class
```

Stability Report Form

```
Public Class StabilityReport

    Private Sub StabilityReport_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        ForcesGridView.DataSource = ResultForcesTable
        StabilityGridView.DataSource = SafetyFactorTable
        StabilityParametersView.DataSource = StabilityParametersTable

        StabilityParametersView.Columns(0).Width = 245
        StabilityParametersView.Columns(1).Width = 95
        StabilityParametersView.DefaultCellStyle.WrapMode = DataGridViewTriState.True
        StabilityParametersView.Columns(2).Width = 65

        ForcesGridView.DefaultCellStyle.WrapMode = DataGridViewTriState.True
        StabilityGridView.DefaultCellStyle.WrapMode = DataGridViewTriState.True
    End Sub
```