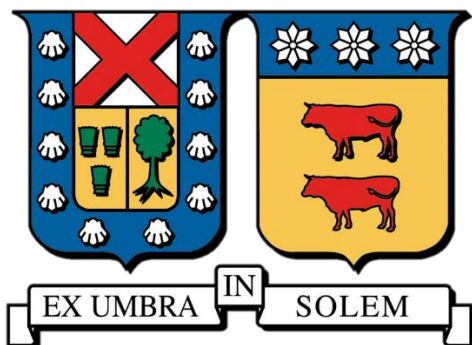


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE INGENIERÍA METALÚRGICA Y DE MATERIALES

SANTIAGO-CHILE



**MODELO DE DILUCIÓN POR MÉTODO VOLUMÉTRICO DE
LAUBSCHER**

IGNACIO ALBERTO SÁNCHEZ SAAVEDRA

MEMORIA PARA OPTAR AL TÍTULO PROFESIONAL DE

INGENIERO CIVIL DE MINAS

PROFESOR GUÍA:

VICTOR ENCINA MONTENEGRO

2024

Santiago, Chile

Contenido

Resumen:	4
Introducción:	5
Objetivos:	6
Objetivos Generales:	6
Objetivos específicos:	6
Revisión bibliográfica:	7
Minería Subterránea	7
Block Caving:	7
Software PC-BC en la Planificación Minera:	7
Características y Beneficios del Software PC-BC:	8
Dilución:	8
Método de Laubscher:	9
Clasificación MRMR	9
Factores de Ajuste	9
Aplicaciones del Método de Laubscher	9
Modelo Gráfico de Laubscher:	10
Descripción del Gráfico	11
Extracción Inicial.....	12
Propagación del Hundimiento.....	12
Extracción Subsecuente	12
Extracción Final y Mezcla Vertical	13
Metodología	14
Variables de entrada al proyecto	14
Validación del Programa	15
Desarrollo	15
Código.....	15
Análisis Crítico de las Limitaciones de la Metodología	18
Limitaciones de la metodología:	18
Resultados:	18
Caso para PED= 10%.....	19
Caso para PED= 80%.....	20
Comparación de Resultados.....	20

Ley Promedio Comparativa	21
Análisis de Resultados y Validación del Programa	21
Conclusiones	23
Referencias	24
Anexo	25
Código 1 - Sección 1	25
Código 1 - Sección 2	25
Código 1 - Sección 3	26
Código 1 - Sección 4	26
Código 1 - Sección 5	26
Código 1 - Sección 6	27
Código 1 - Sección 7	28
Código 1 - Sección 8	30
Código 1 - Ejecución del flujo completo.....	32

Resumen:

En esta tesis se aborda la problemática de la escasez de software especializado y asequible para la planificación y operación de minas que utilizan el método de Block Caving. El objetivo principal de esta investigación es desarrollar un algoritmo de cálculo de dilución por método volumétrico de Laubscher, implementado en Python, para generar un programa que facilite la planificación minera en operaciones de Block Caving.

El trabajo realizado incluye la implementación de un modelo de selección de reservas no diluidas, la generación de un algoritmo de cálculo de ponderadores y la aplicación de este algoritmo para obtener un modelo diluido de bloques. Los datos utilizados provienen de un archivo CSV geológico, con variables espaciales, el valor del bloque y la ley del mineral. Se emplearon técnicas de integración simbólica y suma acumulativa para determinar el piso óptimo y calcular las leyes modificadas.

El programa desarrollado se validó mediante pruebas comparativas con Microsoft Excel, utilizando dos niveles de dilución (10% y 80%). Los resultados mostraron una alta precisión y coincidencia entre el programa y Excel, confirmando la fiabilidad del algoritmo implementado. El software permite una gestión eficiente de la dilución y la planificación de la secuencia de extracción, siendo económico y accesible, especialmente dirigido a estudiantes.

Este proyecto representa un avance significativo en la accesibilidad tecnológica para la planificación minera, ofreciendo una herramienta gratuita y eficaz que puede transformar el aprendizaje y la práctica en operaciones de Block Caving. La propuesta fomenta una mayor inclusión tecnológica en el sector minero, proporcionando una base sólida para futuros desarrollos y mejoras en la planificación y operación de minas subterráneas.

Introducción:

La minería subterránea representa un desafío constante debido a la necesidad de equilibrar la viabilidad económica y la seguridad operativa, especialmente cuando el ratio de estéril mineral implica costos de explotación elevados. Entre las técnicas subterráneas, el método de Block Caving ha emergido como una opción viable para la extracción de cuerpos de minerales de baja ley, destacándose por su alta capacidad de producción y eficiencia operativa. Sin embargo, uno de los retos más significativos en la implementación de este método es la gestión de la dilución, que puede afectar negativamente la rentabilidad y la calidad del mineral extraído.

Actualmente, los softwares especializados en planificación minera, como el PC-BC, ofrecen soluciones avanzadas, pero a menudo inaccesibles para operaciones con recursos financieros limitados debido a sus altos costos. Esta situación resalta la necesidad de desarrollar herramientas más asequibles y accesibles que puedan ser utilizadas tanto por profesionales de la minería como por estudiantes en formación.

En este contexto, la presente tesis tiene como objetivo principal generar un algoritmo de cálculo de dilución utilizando el método volumétrico de Laubscher, implementado en el lenguaje de programación Python. El desarrollo de este algoritmo no solo busca proporcionar una herramienta eficiente para la planificación minera, sino también democratizar el acceso a tecnologías avanzadas, facilitando el aprendizaje y la experimentación en el campo de la minería subterránea.

La metodología empleada combina el análisis de datos geológicos, técnicas de integración simbólica y modelos de suma acumulativa para lograr una solución integral y precisa. Al ofrecer una alternativa económica y efectiva a los softwares comerciales, esta investigación pretende contribuir al avance tecnológico en la minería subterránea, promoviendo una mayor inclusión y accesibilidad a herramientas de planificación avanzadas.

Objetivos:

Objetivos Generales:

- El objetivo principal de este proyecto es generar un algoritmo de cálculo de dilución por método volumétrico de Laubscher, a través de la utilización del lenguaje de programación Python.

Objetivos específicos:

- Implementar el modelo de selección de reservas no diluidas.
- Generar un algoritmo de cálculo de ponderadores.
- Aplicar el algoritmo para generar un programa de cálculo de reservas diluidas.

Revisión bibliográfica:

Minería Subterránea

Cuando el ratio de estéril mineral implica un costo de explotación que supera la viabilidad económica, es necesario optar por una técnica de extracción subterránea. La elección del sistema de explotación subterránea se determina en base a variables como el tipo de roca, mineralización, parámetros económicos, entre otros. (Harmin, 2001) Los métodos de explotación se subdividen en tres grandes grupos:

- Autosoportados
- Artificialmente soportados
- No soportados

Block Caving:

Dentro de los métodos no soportados se encuentra el método Block Caving. Es una técnica significativa en el ámbito minero, especialmente para la extracción de cuerpos de minerales de baja ley en minería subterránea. Este método es frecuentemente reconocido como una opción viable para la minería subterránea en masa. (Tegachouang & Bowa, 2022). El enfoque del método Block Caving se basa en usar una combinación de la fuerza de gravedad y las tensiones internas de la roca para fragmentar y descomponer la masa rocosa en piezas manejables (Pourrahimian & Askari-Nasab, 2010).

Las ventajas que presenta el método de Block Caving se incluyen una alta capacidad de producción, mínimo desarrollo de explotación, alta productividad laboral, bajo costo y notoria seguridad operacional. (Fengyu & Kuangdi, 2023). Además, con el paso del tiempo, Block Caving ha evolucionado para mejorar su viabilidad, seguridad, eficiencia, respaldado por diversas investigaciones y prácticas dirigidas que buscan optimizar el tiempo de ejecución y la inversión de capital. (Melati, y otros, 2023)

Software PC-BC en la Planificación Minera:

La planificación minera, en particular cuando se utiliza el método de Block Caving, es un ejercicio técnico y sofisticado que exige un manejo preciso de los recursos y una optimización constante de los procesos para generar una extracción eficiente y segura (Paravarzar, Askari-Nasab, Pourrahimian, & Emery, 2020). Por ello, la implementación de un software especializado se ha

vuelto esencial para lograr una gestión eficaz. Entre los más destacados en este campo es el software PCBC de GEOVIA, diseñado específicamente para optimizar la planificación de minas que operan bajo el método de Block Caving. (Khodayari & Pourrahimian, 2019)

Características y Beneficios del Software PC-BC:

El software PC-BC proporciona una robusta plataforma para la generación de horarios de producción optimizados. Además, permite modelar y comparar rápidamente diferentes configuraciones de mina, lo que es crucial para optimizar operaciones mineras (Diering, 2000). También facilita la modelación eficaz de la mezcla de material dentro de las columnas de extracción y el análisis de migración de material tanto vertical como horizontalmente. Además, simula la extracción de material en cada punto de extracción designado en la mina para todos los periodos operativos, asegurando una extracción acorde a las características geotécnicas del yacimiento.

Una de las características de este software es brindar a los ingenieros una herramienta especializada en la planificación y diseño de operaciones de Block Caving. Está diseñado específicamente para facilitar las fases de prefactibilidad, permitiendo comparaciones detalladas de diferentes disposiciones mineras y optimizando las operaciones basadas en datos precisos sobre las calificaciones y tasas de desarrollo. Su principal objetivo principal es acelerar y mejorar el proceso de diseño y planificación en operaciones de Block Caving, entregando un análisis rápido y detallados que ayuden a tomar decisiones cruciales (Diering, 2000).

A pesar de que el software PC-BC, tenga muchos beneficios, este presenta un gran desafío para la industria minera, el cual es el costo. Debido a la escasez y exclusividad de estos softwares en el mercado, por lo que su precio puede ser una barrera significativa para la mediana y pequeña minería.

Dilución:

La dilución en minería es un concepto esencial que hace referencia al material de desecho (material estéril o con una ley muy baja) que se mezcla con el mineral extraído. Esto aumenta el tonelaje que debe ser procesado y reduce la ley del mineral. Este problema es especialmente relevante en métodos de minería subterránea como el "block caving". Existen varios tipos de dilución: interna, externa y pérdida de mineral. La dilución puede definirse matemáticamente como la relación entre el tonelaje de material estéril extraído y el tonelaje total extraído (incluyendo tanto el mineral como el material estéril).

Método de Laubscher:

El método de Laubscher es una técnica empírica utilizada en la minería subterránea para evaluar la estabilidad de las excavaciones y diseñar sistemas de soporte apropiados. Este enfoque se basa en la clasificación de masa de roca (MRMR) y proporciona una guía práctica para determinar las dimensiones y el soporte necesarios para mantener la estabilidad de las estructuras subterráneas (Laubscher, 1990).

Clasificación MRMR

La MRMR es una medida ajustada de la calidad de la roca que considera varios factores geomecánicos. Se deriva de la clasificación original de calidad de roca (RQD), modificada según las condiciones específicas del terreno y el entorno minero. Los ajustes incluyen la orientación de las discontinuidades, el tamaño del bloque, la condición de las superficies de discontinuidad y la influencia del agua (Laubscher, 1990) (Bieniawski, 1989).

Factores de Ajuste

1. **Orientación de las Discontinuidades:** Las fracturas y fallas afectan la estabilidad de la masa rocosa. El método de Laubscher ajusta la RQD según la orientación de estas discontinuidades en relación con la excavación (Laubscher, 1990).
2. **Tamaño del Bloque:** Se considera el tamaño de los bloques de roca delimitados por las discontinuidades. Bloques más grandes generalmente implican mayor estabilidad (Laubscher, 1990) (Bieniawski, 1989).
3. **Condición de las Superficies de Discontinuidad:** La condición de las superficies de discontinuidad (rugosidad, llenado, etc.) influye en la estabilidad. Superficies rugosas y sin relleno son más estables (Laubscher, 1990).
4. **Influencia del Agua:** La presencia de agua puede debilitar la masa rocosa. La RQD se ajusta según el grado de saturación de la roca y la presión del agua en las discontinuidades (Laubscher, 1990) (Bieniawski, 1989).

Aplicaciones del Método de Laubscher

1. **Diseño de Soporte:** El método de Laubscher proporciona directrices para el diseño de sistemas de soporte en minas subterráneas. Basado en la MRMR, se pueden determinar los tipos y dimensiones de soportes necesarios, como pernos de anclaje y marcos de acero (Laubscher, 1990).
2. **Evaluación de la Estabilidad:** Utilizando la clasificación MRMR, los ingenieros pueden evaluar la estabilidad de cavidades y taludes subterráneos, identificando áreas que requieren refuerzo adicional (Laubscher, 1990).
3. **Optimización del Diseño Minero:** Al proporcionar una evaluación cuantitativa de la calidad de la roca y su estabilidad, el método de Lau (Laubscher, 1990)bscher ayuda a optimizar el diseño de la mina, reduciendo riesgos y mejorando la eficiencia operativa (Laubscher, 1990) (Bieniawski, 1989).

Modelo Gráfico de Laubscher:

Laubscher propone un gráfico que permite determinar la ley de producción en cada etapa del tiraje, como se muestra en la figura 1. Para ello, es necesario conocer la distribución de leyes en la columna de extracción. El tonelaje que se puede producir económicamente en cada punto de extracción depende de:

1. El tonelaje y la ley: Tanto del mineral como del estéril, a lo largo de la columna de tiraje (su distribución).
2. El momento de dilución: El momento a partir del cual se tendrá dilución en los puntos de extracción.

Es evidente que, si la ley del mineral es alta en la zona de contacto con el estéril, se puede aceptar una cantidad significativa de dilución. Del mismo modo, si la ley del diluyente en la zona de contacto no difiere mucho de la ley de corte (ley de cierre de los puntos de extracción), se puede aceptar una cantidad mayor de dilución. Por el contrario, si la ley del mineral de contacto es baja y la dilución pobre, solo se podrá aceptar una cantidad mínima de dilución. En la figura 1, se puede observar que la composición del material en los puntos de tiraje depende del porcentaje de extracción. (De los Ríos Juris, 1971)

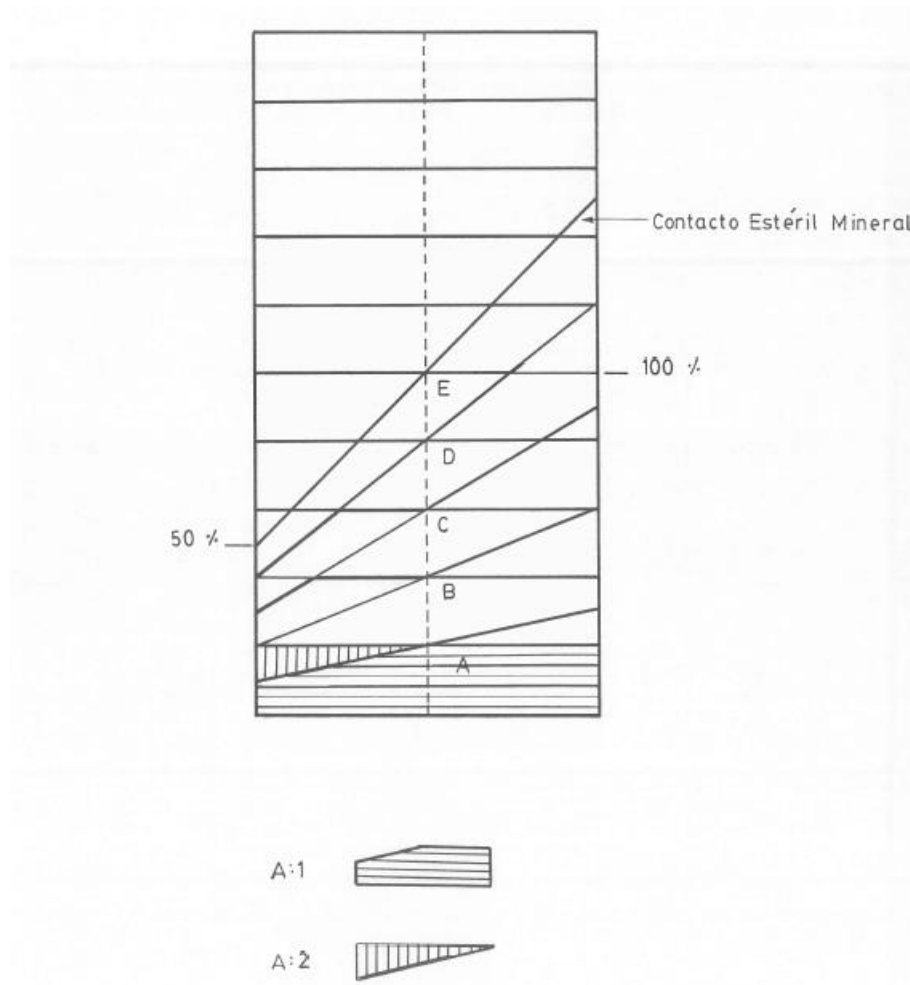


Figura 1: Modelo gráfico de Laubscher para determinar leyes de extracción (De los Ríos Juris, 1971).

Descripción del Gráfico

Según De los Ríos Juris (1971), las líneas horizontales en el gráfico muestran el porcentaje de extracción, con el tonelaje total de mineral disponible en la línea del 100%, incluyendo la dilución. Las líneas inclinadas definen las zonas con diferentes leyes de mineral en la columna de extracción.

La composición del material en el punto de extracción se determina por la intersección de las líneas inclinadas y horizontales. Para simplificar, Laubscher sugiere que la línea de contacto entre mineral y estéril sea recta, pasando por el punto medio de la horizontal del 100% de extracción. Cuanto más temprano sea el ingreso de la dilución, más inclinada será esta línea respecto a la horizontal.

En los bloques explotados por hundimiento, la pendiente de la línea de contacto aumenta con la dilución antes del 100% de extracción del túnel. El mineral desplazado por la dilución solo se puede

recuperar mediante sobretiro. Cuanto más temprano sea el ingreso de la dilución, más inclinada será la línea de contacto y mayor será el volumen de mineral desplazado.

Con un 50% de dilución, se puede configurar la distribución de las líneas inclinadas que muestran los contactos de diferentes leyes, permitiendo estimar la ley de producción para cualquier porcentaje de extracción. Estas líneas determinan la ley para un 20% de extracción y también indican el tonelaje de mineral desplazado.

Asignando valores de leyes A, B, C, D y E para cada capa, la ley se calcula con la Ecuación 1:

$$Ley = \frac{A \cdot A1 + B \cdot A2}{(A1 + A2)} \quad (1)$$

Extracción Secuencial de Columnas

Extracción Inicial

La extracción inicial de un porcentaje del mineral en una columna se realiza para iniciar el colapso y asegurar un flujo controlado del material. Esta fase inicial puede involucrar la extracción de aproximadamente un 20-30% del volumen total de la columna. Esta extracción parcial crea un espacio vacío que facilita el hundimiento y el colapso del mineral desde las zonas superiores (Brown, 2003).

Propagación del Hundimiento

Tras la extracción inicial, se permite que el hundimiento se propague hacia arriba a medida que el mineral colapsa en el espacio creado. Este proceso debe ser monitoreado cuidadosamente para asegurar que el colapso sea uniforme y controlado. Se utilizan tecnologías avanzadas de monitoreo sísmico y vigilancia de la subsidencia para garantizar la estabilidad de la mina durante esta fase (Brown, 2003).

Extracción Subsecuente

Después de que el hundimiento inicial se ha estabilizado, se lleva a cabo una segunda fase de extracción. En esta etapa, se puede extraer un porcentaje adicional del mineral, aprovechando el colapso y la fragmentación causados por la primera extracción. Este enfoque secuencial permite una gestión más eficiente del material y reduce la dilución del mineral con material estéril (Brown, 2003).

Ley del Mineral

- **Prioridad a Áreas de Alta Ley:** Las áreas con leyes más altas de mineral se priorizan para maximizar la rentabilidad, siempre y cuando no comprometan la estabilidad de la mina. La extracción de material de alta ley debe equilibrarse con la capacidad de procesamiento de la planta para evitar cuellos de botella y asegurar un flujo constante de mineral procesable (Brown, 2003).

Capacidad de Procesamiento

- **Tasa de Hundimiento y Capacidad de la Planta:** La tasa de hundimiento deseada y la capacidad de procesamiento de la planta son factores críticos en la asignación de cuotas de extracción. La capacidad de la planta determina cuánto material se puede procesar en un período dado, lo que a su vez influye en la tasa de extracción y en la secuencia de los puntos de extracción (Brown, 2003).

Durante la simulación de la minería, se verifica si la rebanada inferior contiene suficiente material para cumplir con los requisitos de tonelaje del periodo considerado. Si no es suficiente, todo el material de la rebanada debe extraerse, agotándose así la rebanada y moviendo las restantes hacia abajo para su mezcla vertical. Este proceso asegura una extracción continua y uniforme del mineral (Brown, 2003).

Extracción Final y Mezcla Vertical

La extracción final se realiza para remover el material restante de la columna. Durante esta fase, se asegura que todo el material de la columna se extraiga y se mezclen verticalmente las rebanadas para mantener un flujo uniforme de mineral hacia los puntos de extracción. Este proceso asegura que no queden pilares de mineral sin extraer, lo que podría comprometer la estabilidad de la mina (Brown, 2003).

Metodología

Variables de entrada al proyecto

Los datos utilizados para crear este proyecto provienen de un archivo CSV geológico. Este archivo contiene información crucial que incluye variables espaciales, el valor del bloque y la ley del mineral. Estas variables son fundamentales para lograr los objetivos establecidos en esta memoria, que se centra en desarrollar un modelo diluido de la distribución de minerales.

Descripción de los Datos:

El archivo CSV geológico incluye varios campos importantes:

- **Variables espaciales:** Estas variables permiten ubicar geográficamente cada bloque dentro del yacimiento. Incluyen coordenadas Norte, Este y Cota, que indican la posición tridimensional de los bloques. Estas coordenadas son esenciales para la precisión en la modelación y la planificación de la extracción, ya que permiten identificar exactamente dónde se encuentran los recursos minerales.
- **Valor del bloque valorizado (VB):** Este valor representa la importancia económica de cada bloque en función de la cantidad y calidad del mineral contenido. Es una medida crucial para determinar qué bloques son económicamente viables para la extracción.
- **Ley del mineral:** La ley representa la concentración de mineral valioso presente en cada bloque. Este dato es crucial para evaluar la viabilidad económica de la extracción, ya que una mayor ley suele traducirse en una mayor rentabilidad. Además, la ley es fundamental para lograr el objetivo principal de este proyecto, que es obtener un modelo diluido.

Las variables antes mencionadas deben ser ingresadas a partir de la base de datos, la cual debe poseer el nombre de “Modelo” y su extensión debe ser “.csv”, además de poseer el delimitador “,”.

Un ejemplo del CSV es el siguiente:

El archivo CSV geológico que utilizamos para este proyecto contiene múltiples columnas que detallan las variables necesarias para construir el modelo diluido. Cada fila del archivo representa un bloque específico dentro del yacimiento, con sus correspondientes valores y características. A continuación, se muestra una representación simplificada de cómo se estructuran estos datos en el archivo (ver Tabla 1).

Tabla 1: Ejemplo de estructura de datos en el archivo CSV geológico.

Este	Norte	Cota	VB	Ley
-130	380	1650	-77	0
210	620	1570	104	0,966
330	820	1310	623	1,115
450	640	1330	124	1,489

Validación del Programa

Para validar el programa se debe llevar a cabo el siguiente procedimiento: Primero, se debe realizar la toma de una columna específica del archivo CSV geológico de manera manual. Luego, se comparan los valores de esta columna con los resultados generados por el programa, permitiendo verificar la precisión y consistencia de los datos procesados en relación con los datos originales. A continuación, se debe cambiar la variable PED (Porcentaje de Entrada de Dilución) en el archivo CSV y correr nuevamente el programa con la variable PED modificada. Después, se realiza una nueva validación manual similar a la descrita anteriormente para asegurarse de que los resultados sean consistentes y precisos tras el cambio de la variable.

Desarrollo

Como se mencionó en el capítulo anterior, se mostraron las variables que se utilizan en este modelo de bloques en formato CSV. La idea de este programa es obtener un modelo de bloques diluido mediante el modelo de Laubscher. A continuación, se presenta el código paso a paso, explicando cada función utilizada. Para más detalles, consulte el código en el anexo, denominado "Código 1".

Como se mencionó en el capítulo anterior, se mostraron las variables que se utilizan en este modelo de bloques en formato CSV. La idea de este programa es obtener un modelo de bloques diluido mediante el modelo de Laubscher. A continuación, se presenta el código paso a paso, explicando cada función utilizada. Para detalles adicionales, consulte el código en el anexo, denominado "Código 1".

Código

1. Definición de Variables y Funciones

En la primera sección del código (Código 1 - Sección 1), se definen las variables y las funciones básicas necesarias para llevar a cabo los cálculos. Utilizamos las librerías pandas para el manejo de datos y sympy para realizar las integraciones simbólicas necesarias.

2. Función calcular_area_doble

La función `calcular_area_doble(m, b, H)` se encarga de calcular el área bajo una curva definida por la ecuación de una recta y los límites de integración. Esta función es crucial para determinar el área de los bloques, para luego poder hacer la ponderación por área y obtener la ley diluida. Los parámetros de entrada incluyen la pendiente m , la ordenada al origen b , y la altura H . Dependiendo de la posición de la recta con respecto al rectángulo definido por H , la función ajusta

los límites de integración y calcula el área correspondiente utilizando integración doble. Puede encontrar más detalles en el código en el anexo (Código 1 - Sección 2).

3. Función cargar_datos

La función `cargar_datos()` carga los datos desde un archivo CSV y los ordena según las columnas Este, Norte y Cota. Es importante ordenar los datos para facilitar el procesamiento y análisis subsecuente. La función imprime las columnas originales para confirmar que los datos se han cargado correctamente. Luego, los datos se ordenan para asegurar que las operaciones acumulativas se realicen de manera secuencial y coherente. Consulte el anexo para más información (Código 1 - Sección 3).

4. Función suma_acumulativa

La función `suma_acumulativa(df)` calcula la suma acumulativa de la columna VB para cada grupo definido por las columnas Este y Norte. Esto permite acumular los valores de VB a medida que se recorren los datos ordenados, lo que es esencial para identificar los puntos de máximo valor acumulado en etapas posteriores del proceso. Detalles adicionales se encuentran en el anexo (Código 1 - Sección 4).

5. Función indice_optimizador_global

La función `indice_optimizador_global(df)` determina el piso óptimo global, que es el punto donde la suma acumulativa total de VB es máxima. Esta función primero calcula la suma total acumulada y luego encuentra el índice donde esta suma es máxima. Utilizando este índice, se obtiene la cota correspondiente que representa el piso óptimo global. Este punto es crucial para definir las bases de las columnas que se analizarán. Más información está disponible en el código del anexo (Código 1 - Sección 5).

6. Función suma_desde_piso_optimo_hasta_techo

La función `suma_desde_piso_optimo_hasta_techo(df, piso_optimo)` calcula los techos y las sumas acumuladas desde el piso óptimo hasta el techo para cada grupo de datos definido por Este y Norte. Se utiliza una constante adicional para ajustar los valores de las sumas acumulativas; esta constante es el coste por columna, que para este caso se decidió que fuera de 4000. Se identifican los techos donde la suma acumulativa ajustada es máxima. Los resultados se almacenan en diccionarios para facilitar su acceso en etapas posteriores del proceso. Consulte el código en el anexo para más detalles (Código 1 - Sección 6).

7. Función crear_diccionarios

La función `crear_diccionarios(df, piso_optimo, techos, PED)` crea diccionarios de columnas y columnas modificables con bloques de datos filtrados y ajustados según el piso óptimo y el techo. Cada bloque incluye información sobre su ubicación, cota y ley. Los bloques modificables se ajustan considerando un porcentaje de entrada de dilución (PED), creando dos alturas ajustadas (HD y HP) para cada bloque. Estos diccionarios son esenciales para la transformación y el cálculo de la ley modificada en la siguiente etapa. Puede encontrar más detalles en el código en el anexo (Código 1 - Sección 7).

8. Función tamaño_de_bloque

La función `tamaño_de_bloque(df)` calcula el tamaño del bloque en las direcciones X, Y y Z (Este, Norte y Cota) en función de las coordenadas adyacentes del modelo. Esta función recorre el DataFrame para identificar la diferencia en cada coordenada, de manera que pueda obtener el tamaño del bloque en cada dirección. Esto es esencial para definir con precisión el volumen de cada bloque, lo que afecta directamente a los cálculos de ponderación y dilución en el modelo. Más información está disponible en el anexo (Código 1 - Sección 8).

9. Función `transformar_y_calcular_ley`

La función `transformar_y_calcular_ley(columnas, columnas_modificables)` transforma los datos de las columnas y calcula la ley modificada para cada bloque. Utiliza integrales dobles para calcular las áreas bajo las curvas definidas por los bloques modificables y ajusta estas áreas según las leyes de los bloques. La ley modificada se calcula ponderando las áreas ajustadas por las leyes de los bloques modificables. Los resultados parciales se almacenan en una lista que contiene la ubicación, cota, ley original y ley modificada de cada bloque. Más información está disponible en el anexo (Código 1 - Sección 9).

Ejecución del Flujo Completo

Para ejecutar el flujo completo del programa, se siguen los siguientes pasos:

1. **Cargar los Datos:** Se cargan y ordenan los datos desde un archivo CSV utilizando la función `cargar_datos`. Para más información, consulte el anexo (Código 1 - Sección 3).
2. **Calcular la Suma Acumulativa:** Se calcula la suma acumulativa de VB para cada grupo utilizando la función `suma_acumulativa`. Más detalles están disponibles en el anexo (Código 1 - Sección 4).
3. **Determinar el Piso Óptimo Global:** Se encuentra el piso óptimo global donde la suma acumulativa total es máxima utilizando la función `indice_optimizador_global`. Para más detalles, consulte el anexo (Código 1 - Sección 5).
4. **Calcular Techos y Sumas Acumuladas:** Se calculan los techos y sumas acumuladas desde el piso óptimo hasta el techo para cada grupo utilizando la función `suma_desde_piso_optimo_hasta_techo`. Más información está disponible en el anexo (Código 1 - Sección 6).
5. **Crear Diccionarios de Columnas:** Se crean diccionarios de columnas y columnas modificables utilizando la función `crear_diccionarios`. Consulte el anexo para más detalles (Código 1 - Sección 7).
6. **Calcular el Tamaño del Bloque:** La función `tamaño_de_bloque` calcula el tamaño de los bloques en las tres dimensiones (X, Y, Z). Esto permite ajustar el análisis de acuerdo con las especificaciones dimensionales de cada bloque, asegurando precisión en los cálculos de dilución y ponderación. Consulte el anexo para más detalles (Código 1 - Sección 8).
7. **Transformar y Calcular la Ley Modificada:** Se transforman los datos y se calcula la ley modificada para cada bloque utilizando la función `transformar_y_calcular_ley`. Más información está disponible en el anexo (Código 1 - Sección 9).

Finalmente, los resultados se guardan en un archivo CSV y se imprimen.

El flujo completo del programa es ajustable, por ejemplo, el valor del porcentaje de entrada de dilución (PED) puede ser modificado según las necesidades específicas del análisis. El programa

permite obtener un modelo de bloques diluido de manera detallada y precisa, facilitando la toma de decisiones en la gestión y planificación minera.

Análisis Crítico de las Limitaciones de la Metodología

Limitaciones de la metodología:

1. **Dependencia de la calidad de los datos:** La precisión del modelo depende de la calidad y exactitud de los datos geológicos proporcionados. Cualquier error o inconsistencia en los datos de entrada puede afectar significativamente los resultados.
2. **Suposiciones en el modelo de Laubscher:** El modelo de Laubscher, aunque ampliamente aceptado, se basa en suposiciones empíricas que pueden no aplicarse perfectamente a todas las situaciones geológicas. Esto puede limitar la precisión de los resultados en ciertos contextos.
3. **Capacidades computacionales:** El procesamiento de grandes conjuntos de datos geológicos y la realización de cálculos complejos puede requerir recursos computacionales significativos. Las limitaciones en la capacidad de procesamiento pueden afectar la velocidad y eficiencia del programa.

Resultados:

Para evaluar la precisión del programa desarrollado para un método de dilución, se realizó una prueba comparativa utilizando Microsoft Excel con el fin de verificar si ambos métodos entregan los mismos resultados. Esta evaluación se llevó a cabo utilizando dos niveles de dilución: PED 10% y PED 80%. En ambos casos, se tomó como base la misma columna de datos para asegurar la consistencia en la comparación.

Coordenadas de la columna:

- **Este:** -130
- **Norte:** 780
- **Altura de columna:** 200

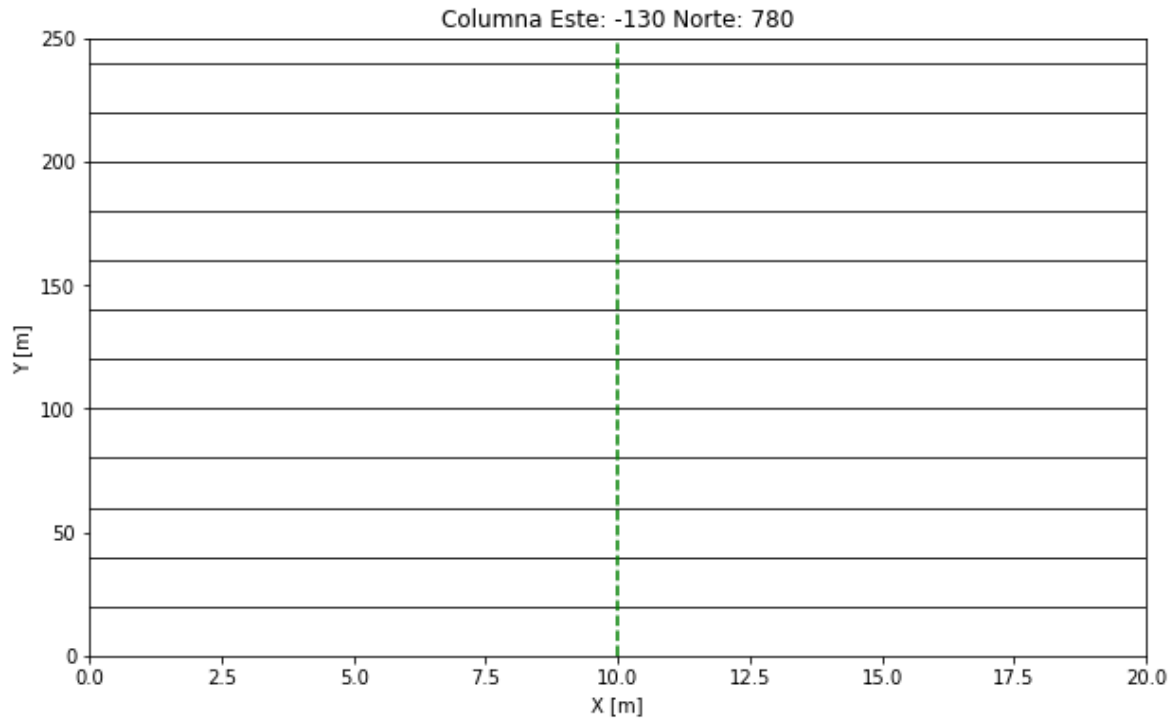


Gráfico 1: Columna para realizar las pruebas.

Caso para PED= 10%

En la siguiente imagen, se graficaron cada una de las rectas que pasan por la columna en color rojo. Siguiendo el método de Laubscher, basado en la ley de la tajada, se utilizó el área como ponderador para luego determinar la ley total.

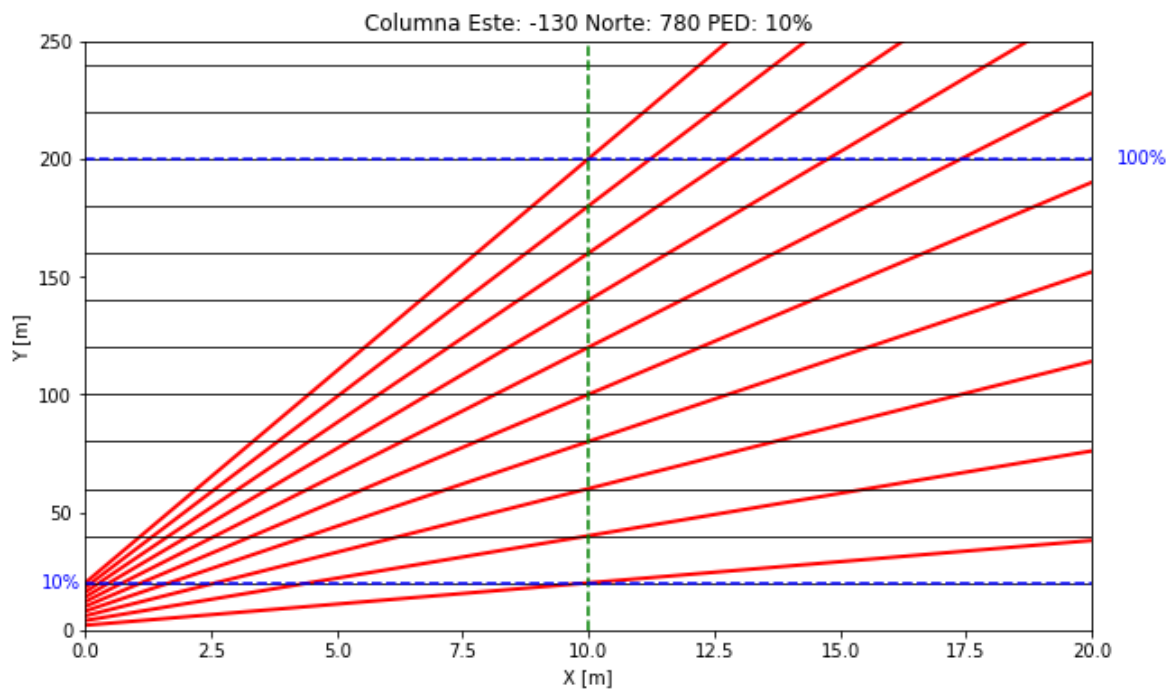


Gráfico 2: Columna con las respectivas rectas para PED 10%.

En este caso, los datos fueron ingresados tanto en el programa como en Excel, y los resultados obtenidos fueron analizados para identificar cualquier discrepancia. Los valores calculados por el programa coincidieron con los resultados generados por Excel, confirmando la precisión del programa para este nivel de dilución.

Caso para PED= 80%

Para el nivel de dilución PED 80, se siguió el mismo procedimiento utilizado en PED 10%. En la siguiente imagen se graficaron las rectas correspondientes, también en color rojo. Nuevamente, se aplicó el método de Laubscher utilizando el área como ponderador para determinar la ley total.

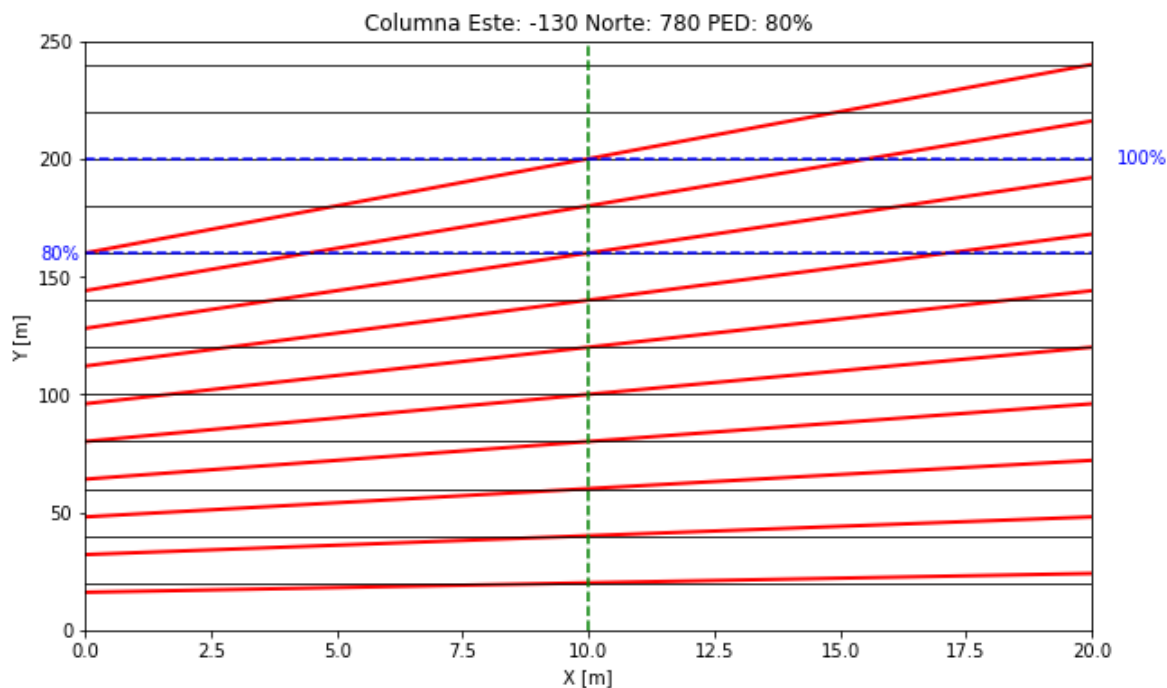


Gráfico 3: Columna con las rectas correspondientes a PED 80%.

Los datos fueron ingresados en el programa y en Excel, y se analizaron los resultados para identificar cualquier discrepancia. Al igual que con PED 10%, los valores calculados por el programa coincidieron con los resultados generados por Excel. Esto confirma la precisión del programa para este nivel de dilución.

Comparación de Resultados

A continuación, se presentan las tablas comparativas de los resultados obtenidos tanto en el programa como en Excel para ambos niveles de dilución:

Tabla 2: Comparación de resultados para PED 10%.

Este	Norte	Cota	Programa	Excel
-130	780	1410	0,568494	0,568243
-130	780	1430	0,708464	0,708125
-130	780	1450	0,838321	0,837952

Tabla 3: Comparación de resultados para PED 80%.

Este	Norte	Cota	Programa	Excel
-130	780	1410	0,5075	0,5075
-130	780	1430	0,56	0,56
-130	780	1450	1,21345	1,21345

Ley Promedio Comparativa

La siguiente tabla presenta la comparación de la ley promedio para los diferentes niveles de dilución y para el estado in-situ (sin dilución). La ley promedio se calcula como el promedio aritmético de todas las leyes individuales en cada caso.

Tabla 4: Comparación de la ley promedio entre los diferentes niveles de dilución y el estado in-situ.

	In-situ	PED 80%	PED 10%
Ley promedio	0,9523	0,9038	0,7447

Análisis de Resultados y Validación del Programa

Precisión del Programa

Para evaluar la precisión del programa desarrollado, se realizaron pruebas comparativas utilizando Microsoft Excel como referencia. Este análisis se llevó a cabo en dos niveles de dilución: PED 10% y PED 80%. Los resultados muestran una excelente coincidencia entre el programa y Excel, lo que sugiere que el programa es preciso y confiable. A continuación, se detallan los puntos clave del análisis:

1. **Coincidencia de Resultados:** En ambos niveles de dilución (PED 10% y PED 80%), los resultados obtenidos del programa y de Excel coincidieron casi perfectamente. Esto se evidenció en las tablas comparativas, donde las diferencias fueron mínimas, indicando una alta precisión del programa.
2. **Método de Laubscher:** Se utilizó el método de Laubscher para determinar la ley total. Este método, basado en la ley de la tajada, es ampliamente reconocido y utilizado en la industria. El hecho de que los resultados del programa coincidan con los obtenidos mediante este método aplicado en Excel refuerza la validez del programa.
3. **Gráficos y Rectas:** Las gráficas de las rectas que pasan por la columna para ambos niveles de dilución (PED 10% y PED 80%) muestran una representación visual coherente con los cálculos realizados. La consistencia en la representación gráfica entre el programa y Excel es lo esperado según los niveles de dilución aplicados, ya que el PED 10% y PED 80% deben reflejar diferentes grados de dispersión y concentración del mineral, acorde con el porcentaje de dilución aplicado.

Validación del Programa

La validación del programa se basa en la comparación de los resultados con los obtenidos mediante Excel, y la confirmación de la consistencia de los cálculos a través de varios niveles de dilución. A continuación, se presenta un análisis más detallado de la validación:

1. **Ley Promedio Comparativa:** La Tabla 4 de comparación de la ley promedio entre diferentes niveles de dilución y el estado in-situ muestra que la mayor ley promedio es la in-situ, lo cual era lo esperado. Además, la ley del PED 80 es más cercana a la de PED 10, lo cual también es consistente con las expectativas, dado que en uno de ellos hay menor dilución que en el otro.
2. **Consistencia en Diferentes Niveles de Dilución:** La precisión del programa en distintos niveles de dilución (10% y 80%) demuestra su robustez y versatilidad. La capacidad de mantener la precisión en diferentes escenarios sugiere que el programa es fiable para su uso en diversos contextos de análisis de dilución.

Conclusiones

En esta memoria se ha investigado la problemática de la escasez de software especializado y asequible para la planificación y operación de minas que utilizan el método de Block Caving. Los softwares actuales, aunque avanzados, presentan costos elevados que limitan su accesibilidad, especialmente para minas con recursos financieros limitados.

El objetivo principal de esta investigación fue desarrollar un software que facilite la planificación minera en operaciones de Block Caving, utilizando un calculador de dilución basado en el método volumétrico de Laubscher. Esta herramienta proporciona rapidez en el procesamiento de datos, genera información utilizable en otros programas y es fácil de usar, requiriendo pocos datos de entrada para diluir el modelo de bloques o planificar la secuencia de extracción.

El software desarrollado ofrece una solución económica y gratuita, dirigida principalmente a estudiantes, facilitando su aprendizaje y experimentación en la planificación minera y promoviendo una mayor inclusión tecnológica en el sector.

A pesar de los logros alcanzados, este proyecto representa solo un primer paso hacia la creación de un ecosistema de software integral para Block Caving. Se recomienda continuar con el desarrollo del software, incorporando funcionalidades adicionales basadas en las necesidades de los usuarios y las tendencias tecnológicas emergentes en la minería. Mejorar la interfaz de usuario para una mayor accesibilidad, adaptar el software a diferentes entornos mineros y desarrollar más componentes que aborden todas las facetas de la planificación y operación de minas representan áreas clave para futuras expansiones.

Aunque los resultados han sido positivos, es importante considerar posibles áreas de mejora. La dependencia de la calidad de los datos geológicos y las suposiciones inherentes al modelo de Laubscher pueden influir en la precisión del programa. Es esencial continuar investigando y refinando estos aspectos para mejorar aún más la fiabilidad del software.

En conclusión, esta memoria ha desarrollado un software innovador y accesible que puede transformar la forma en que los estudiantes aprenden y practican la planificación y operación de minas con el método de Block Caving. Proporcionando una herramienta que combina eficacia, precisión y costo-efectividad, se abre una nueva vía para la educación en minería subterránea. Los resultados y metodologías presentadas en este trabajo deben servir de inspiración y guía para futuros desarrollos en el campo de la ingeniería de minas, contribuyendo así al avance continuo del conocimiento y la tecnología en la industria minera global.

Referencias

- Bieniawski, Z. T. (1989). *Engineering Rock Mass Classifications: A Complete Manual for Engineers and Geologists in Mining, Civil, and Petroleum Engineering*. Wiley-Interscience.
- Brown, E. (2003). *Block Caving Geomechanics*. Brisbane: University of Queensland, JKMR Consulting Group Inc., Itasca Consulting Group Inc.
- De los Ríos Juris, H. (1971). *Apoyo computacional a la planificación en una explotación por panel caving y su aplicación al Proyecto INCA*. Copiapo: Universidad de Atacama, Facultad de Ingeniería, Departamento de Minas.
- Diering, T. (2000). *PC-BC: Block Cave Design and Draw Control System*. Brisbane: MassMin 2000.
- Fengyu, R., & Kuangdi, X. (2023). *Block Caving Mining Method, Application Of*. Singapore: The ECPH Encyclopedia of Mining and Metallurgy.
- Harmin, H. (2001). Underground Mining Methods and applications. En H. Harmin, *Underground Mining Methods: Engineering Fundamentals and International Case Studies* (págs. 3-14). Littleton, Colorado: Society for Mining, Metallurgy and Exploration.
- Khodayari, F., & Pourrahimian, Y. (2019). *Long-term production scheduling optimization and 3D material mixing analysis for block caving mines*. Mining Technology.
- Laubscher, D. H. (1990). A geomechanics classification system for the rating of rock mass in mine design. *Journal of the South African Institute of Mining and Metallurgy*.
- Melati, S., Wattimena, R. K., Sahara, D. P., Syafrizal, Simangunsong, G. M., Hidayat, W., . . . Shinta Felisia, R. R. (2023). *Block Caving Mining Method: Transformation and Its Potency in Indonesia*. Energies.
- Paravarzar, S., Askari-Nasab, H., Pourrahimian, Y., & Emery, X. (2020). *Operational mine planning in block cave mining: a simulation-optimisation approach*.
- Pourrahimian, Y., & Askari-Nasab, H. (2010). *An overview of block caving operation and available methods for production scheduling of block cave mines*. Edmonton, Canada: University of Alberta.
- Tegachouang, N. C., & Bowa, V. M. (2022). *Study of the Influence of Block Caving Underground Mining on the Stability of the Overlying Open Pit Mine*.

Anexo

Código 1 - Sección 1

```
import pandas as pd
```

```
import sympy as sp
```

```
x, y = sp.symbols('x y')
```

Código 1 - Sección 2

```
def calcular_area_doble(m, b, H):
```

```
    # Definir las variables simbólicas
```

```
    x, y = sp.symbols('x y')
```

```
    # Definir los límites de integración en x
```

```
    x_lim_inf = 0
```

```
    x_lim_sup = 20
```

```
    # Calcular los puntos de intersección para determinar si la recta intersecta el rectángulo
```

```
    y_interseccion_0 = b
```

```
    y_interseccion_20 = m * 20 + b
```

```
    integral_doble = 0
```

```
    if y_interseccion_0 < H-20 and y_interseccion_20 < H-20:
```

```
        # Caso donde la recta está completamente por debajo del rectángulo
```

```
        integral_doble = 0
```

```
    elif y_interseccion_0 > H and y_interseccion_20 > H:
```

```
        y_lim_inf = H - 20
```

```
        y_lim_sup = H
```

```
        integral_doble = sp.integrate(1, (y, y_lim_inf, y_lim_sup), (x, x_lim_inf, x_lim_sup))
```

```
    else:
```

```
        if y_interseccion_0 < H-20 and y_interseccion_20 > H-20:
```

```
            x_interseccion_inf = (H - 20 - b) / m
```

```
            x_interseccion_sup = 20
```

```

    x_interseccion_inf = max(x_lim_inf, x_interseccion_inf)
    x_interseccion_sup = min(x_lim_sup, x_interseccion_sup)
else:
    x_interseccion_inf = x_lim_inf
    x_interseccion_sup = x_lim_sup

y_lim_inf = H - 20
y_lim_sup = sp.Min(m * x + b, H)

integral_doble = sp.integrate(1, (y, y_lim_inf, y_lim_sup), (x, x_interseccion_inf,
x_interseccion_sup))

return integral_doble

```

Código 1 - Sección 3

```

def cargar_datos():
    df = pd.read_csv(r'C:\Users\Nacho\Desktop\Modelo de bloques finales\Modelo_bloques.csv',
delimiter=';')

    print("Columnas originales:", df.columns)

    df.sort_values(by=['Este', 'Norte', 'Cota'], ascending=[True, True, True], inplace=True)

    return df

```

Código 1 - Sección 4

```

def suma_acumulativa(df):
    df['Suma_Acumulativa'] = df.groupby(['Este', 'Norte'])['VB'].cumsum()

    return df

```

Código 1 - Sección 5

```

def indice_optimizador_global(df):

```

```

df['Suma_Total'] = df['VB'].cumsum()

idx_max = df['Suma_Total'].idxmax()

piso_optimo_global = df.loc[idx_max, 'Cota']

return piso_optimo_global

```

Código 1 - Sección 6

```
def suma_desde_piso_optimo_hasta_techo(df, piso_optimo):
```

```
    CONSTANTE_ADICIONAL = -4000
```

```
    techos = {}
```

```
    sumas_acumuladas = {}
```

```
    df = df[df['Cota'] >= piso_optimo].copy()
```

```
    grupos = df.groupby(['Este', 'Norte'])
```

```
    for (este, norte), grupo in grupos:
```

```
        grupo = grupo.sort_values(by='Cota', ascending=True)
```

```
        grupo['Suma_Acumulativa'] = grupo['VB'].cumsum()
```

```
        grupo['Suma_Acumulativa'] += CONSTANTE_ADICIONAL
```

```
        idx_max = grupo['Suma_Acumulativa'].idxmax()
```

```
        techo = grupo.at[idx_max, 'Cota']
```

```
        df_filtrado = grupo[(grupo['Cota'] >= piso_optimo) & (grupo['Cota'] <= techo)]
```

```
        suma_acumulada = df_filtrado['VB'].sum() + CONSTANTE_ADICIONAL
```

```
    if suma_acumulada > 0:
```

```
        techos[(este, norte)] = techo
```

```
        sumas_acumuladas[(este, norte)] = suma_acumulada
```

```
    return techos, sumas_acumuladas
```

Código 1 - Sección 7

def crear_diccionarios(df, piso_optimo, techos, PED):

```
    columnas = {}
```

```
    columnas_modificables = {}
```

```
    grupos = df.groupby(['Este', 'Norte'])
```

```
    contador_columna = 1
```

```
    for (este, norte), grupo in grupos:
```

```
        techo = techos.get((este, norte), None)
```

```
        if techo is None:
```

```
            continue
```

```
        altura_columna = techo - piso_optimo + 20
```

```
        if altura_columna <= 0:
```

```
            continue
```

```
        grupo_filtrado = grupo[(grupo['Cota'] >= piso_optimo) & (grupo['Cota'] <= techo)]
```

```
        bloques = []
```

```
        bloques_modificables = []
```

```
        # Generar bloques filtrados por el piso óptimo y el techo
```

```
        for _, row in grupo_filtrado.iterrows():
```

```
            H = row['Cota'] - piso_optimo + 20
```

```
            bloque = {
```

```
                'Este': row['Este'],
```

```
                'Norte': row['Norte'],
```

```
                'Cota': row['Cota'],
```

```
                'Ley': row['Ley'],
```

```
                'H': H
```

```

    }

    bloques.append(bloque)

if not bloques:
    continue

# Calcular HD y HP
num_bloques = len(bloques)
HD_total = altura_columna * PED
HP_total = altura_columna * (1 + (1 - PED))
HD_increment = HD_total / num_bloques
HP_increment = HP_total / num_bloques

HD_current = HD_increment
HP_current = HP_increment

for _ in range(num_bloques):
    bloque_modificable = [HD_current, HP_current, bloques[_]['Ley']]
    bloques_modificables.append(bloque_modificable)
    HD_current += HD_increment
    HP_current += HP_increment

columna_key = f"columna{contador_columna}"
columnas[columna_key] = {
    'bloques': bloques,
    'altura_columna': altura_columna
}

columnas_modificables[columna_key] = bloques_modificables

```

```

        contador_columna += 1

    return columnas, columnas_modificables

```

Código 1 - Sección 8

```

def tamaño_de_bloque(df):

    bloque_z = None

    bloque_y = None

    bloque_x = None


    for i in range(1, len(df)):

        if df.iloc[i]['Cota'] != df.iloc[i-1]['Cota']:

            bloque_z = abs(df.iloc[i]['Cota'] - df.iloc[i-1]['Cota'])

            break


    for i in range(1, len(df)):

        if df.iloc[i]['Norte'] != df.iloc[i-1]['Norte']:

            bloque_y = abs(df.iloc[i]['Norte'] - df.iloc[i-1]['Norte'])

            break


    for i in range(1, len(df)):

        if df.iloc[i]['Este'] != df.iloc[i-1]['Este']:

            bloque_x = abs(df.iloc[i]['Este'] - df.iloc[i-1]['Este'])

            break


    return [bloque_z, bloque_y, bloque_x]

```

Código 1 - Sección 9

```

def transformar_y_calcular_ley(columnas, columnas_modificables):

    resultados = []

```

```
x, y = sp.symbols('x y')
```

```
for key in columnas.keys():
```

```
    bloques = columnas[key]['bloques']
```

```
    bloques_modificables = columnas_modificables[key]
```

```
for bloque in bloques:
```

```
    H = bloque['H']
```

```
    suma_areas_leyes_bloque = 0
```

```
    suma_areas_bloque = 0
```

```
for bloque_modificable in bloques_modificables:
```

```
    HD = bloque_modificable[0]
```

```
    HP = bloque_modificable[1]
```

```
    ley_modificable = bloque_modificable[2]
```

```
    m = (HP - HD) / 20
```

```
    n = HD
```

```
    # Calcular el área bajo la curva usando una integral doble con los límites correctos
```

```
    area = calcular_area_doble(m, n, H)
```

```
    # Restar la suma acumulada de la columna al área calculada
```

```
    area_ajustada = area - suma_areas_bloque
```

```
    # Acumular las áreas ponderadas por la ley del bloque modificable
```

```
    suma_areas_leyes_bloque += area_ajustada * ley_modificable
```

```
    suma_areas_bloque += area_ajustada
```

```

    if suma_areas_bloque == 400:
        # Calcular la ley modificada final para este bloque
        ley_modificada_bloque = suma_areas_leyes_bloque / 400 if suma_areas_bloque != 0
    else 0

    # Agregar ley modificada a los resultados parciales para este bloque
    resultado_parcial = {
        'Este': bloque['Este'],
        'Norte': bloque['Norte'],
        'Cota': bloque['Cota'],
        'Ley': bloque['Ley'],
        'Ley_modificada': ley_modificada_bloque
    }
    resultados.append(resultado_parcial)
    break

return resultados

```

Código 1 - Ejecución del flujo completo

PED = 0.8 # Valor de ejemplo, ajústalo según tus necesidades

```

df = cargar_datos()
df = suma_acumulativa(df)
piso_optimo_global = indice_optimizador_global(df)
techos, sumas_acumuladas = suma_desde_piso_optimo_hasta_techo(df, piso_optimo_global)
columnas, columnas_modificables = crear_diccionarios(df, piso_optimo_global, techos, PED)
resultados = transformar_y_calcular_ley(columnas, columnas_modificables)

# Crear el DataFrame y guardar los resultados en un archivo CSV
df_resultados = pd.DataFrame(resultados)
df_resultados.to_csv('resultados.csv', index=False)

```

```
print("Resultados guardados en resultados.csv")
```