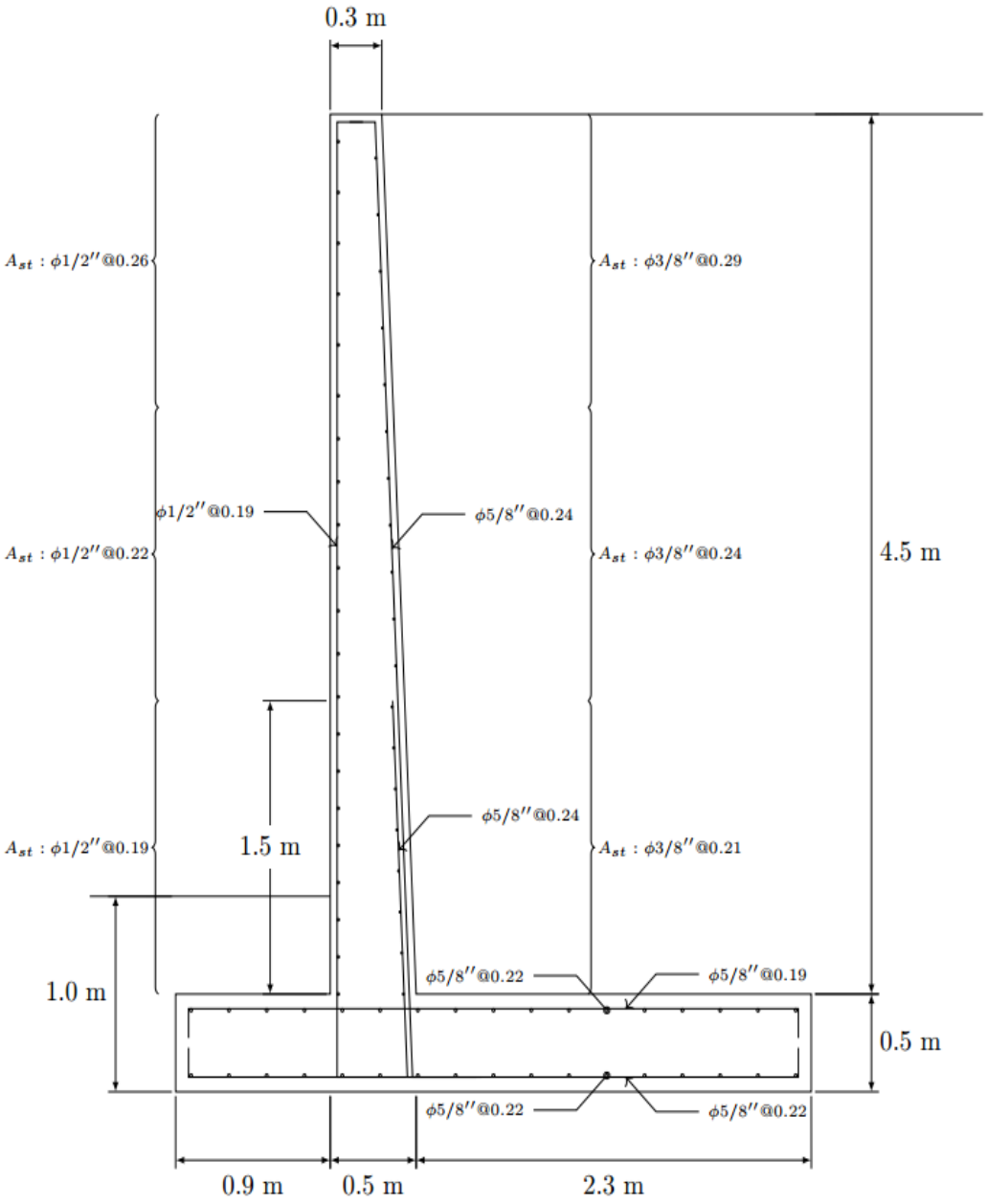
RetWall.jl

v0.1.0

Manual de uso



Índice general

1 Instalación

1.1. Instalando Julia

1.1.1. Windows

1. Descargar el instalador de https://julialang.org/downloads/
2. Ejecutar el archivo y seguir las instrucciones.
3. cuando nos aparezca la ventana mostrada en la figura 1.1, nos aseguraremos que la caja correspondiente a *Add Julia to PATH* este activado.

1.1.2. ubuntu

1. Descargar la última versión de Julia de https://julialang.org/downloads/
2. Extraer en la ubicación deseada.
3. Editar el archivo ~/.bashrc y agregar la siguiente línea:
   export PATH="$PATH:/ruta/a/<carpeta de Julia>/bin"
4. De esta manera se podrá ejecutar Julia desde la línea de comandos

1.2. Instalando RetWall

1. Debemos tener una conexión activa a internet
2. Descargar el paquete de la página de GitHub https://github.com/4lrdyD/RetWall
3. Abrimos Julia
4. Tecleamos `]` para entrar al entorno de manejo de paquetes.
5. Creamos el proyecto tipeando:

```
(@v1.9) pkg> generate ./.julia/packages/RetWall
```

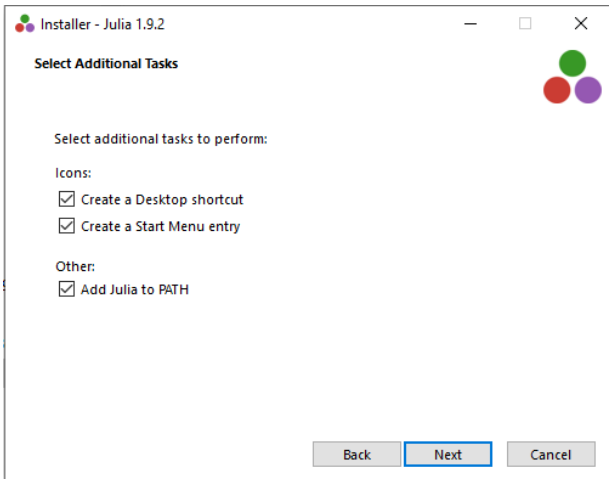


Figura 1.1: Instalación de Julia

6. Con el paso anterior habremos creado la carpeta para alojar el paquete, esta contendrá una subcarpeta llamada *src*, debemos copiar en esta, todo el contenido de la carpeta *src* contenida en el zip descargado, por defecto la carpeta creada estará alojada en:

   - En Windows: *C:\Users\<usuario>\.julia\packages\RetWall*
   - En Ubuntu: */home/<usuario>/.julia/packages/RetWall*

   Cuando nos pregunte si queremos reemplazar el archivo *RetWall.jl* le daremos en aceptar.

7. Volvemos a Julia y tipeamos:

```
(@v1.9) pkg> dev ./.julia/packages/RetWall
(@v1.9) pkg> activate ./.julia/packages/RetWall
(RetWall) pkg> add LinearAlgebra
(RetWall) pkg> activate @v1.9
(@v1.9) pkg> add LinearAlgebra
```

8. Presionamos la tecla retroceso (backspace), para volver a la REPL de Julia

9. tipeamos:

```
using RetWall
```

y ya podremos usar el paquete. De aquí en adelante, cada vez que volvamos a abrir Julia debemos seguir este paso para poder utilizar el paquete.

.

1.3. Instalando LaTeX

1. Para generar los reportes de cálculo, es necesario instalar LaTeX
2. Se recomienda descargar e instalar MikTex de https://miktex.org/download

2 Usando RetWall

2.1. ¿Para que sirve?

RetWall.jl es un paquete para diseño de muros de contención, una de las funcionalidades más importantes es que nos generará una memoria de cálculo automática, para esto es necesario tener instalado LaTeX.
La memoria o reporte de cálculo incluirá un dibujo, de ser el caso, de la distribución de refuerzo propuesta para el muro, por ahora solo está disponible cuando se elija el análisis por la teoría de presión de tierra de Rankine.

2.2. Modo de uso

2.2.1. Primer ejemplo

Las siguientes líneas muestran el diseño estándar completo de un muro de contención típico en voladizo.

```
mywall=typeIwall(b1=.9,t2=.0,t1=.3,t3=.2,b2=2.3,hz=.5,hp=4.5);
orient_model!(mywall.model);
mywall.D=1;
mywall.q=10;
addsoil!(mywall.model,[25 0 16.5]);
addsoil!(mywall.model,[25 13 16.5]);
addmat!(mywall.model,[21000. 0 24]);
RetWall.build_rankine_pline(mywall);
push!(mywall.model.nod,mywall.model.pnod[2:2,:]);
addmat!(mywall.model,[0 0 16.5]);
addmat!(mywall.model,[0. 0 0 420000]);#para el acero de refuerzo
push!(mywall.model.elm,[3 11 10 8 2]);
mywall.model.plinels[1,3]=2;
mywall.model.soilprop[2,5]=166;
report(mywall,design=1,ignore_pasive_moments=1);
```

Las unidades en las que debemos ingresar los valores, será de m (metros) para distancias y de KN (Kilo Newtons) para fuerzas o pesos, las demás unidades deben ser una combinación de estás, por ejemplo, el peso específico deberá estar en KN/m^3, los ángulos deberán ingresarse en grados sexagesimales. Pasaremos a explicar una por una las líneas ejecutadas:

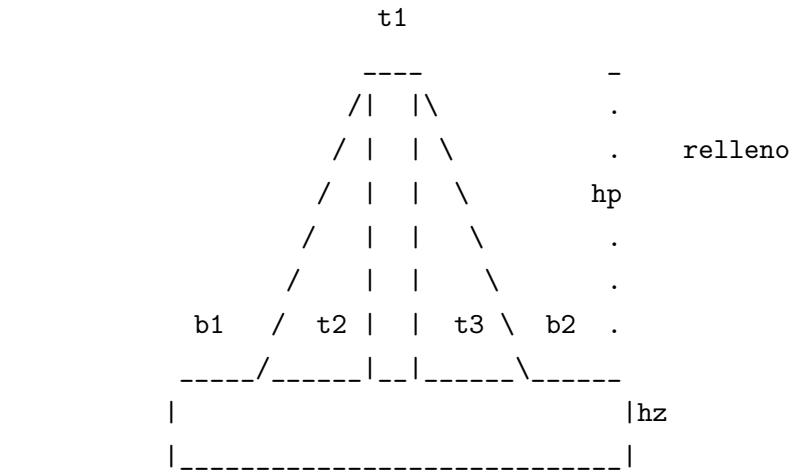
- `mywall=typeIwall(b1=.9,t2=.0,t1=.3,t3=.2,b2=2.3,hz=.5,hp=4.5);`

  luego de ejecutar este comando podemos escribir:

  ```
  mywall
  ```

  que nos mostrará:

```
typeIwall
hp=4.5
hz=0.5
t1=0.3
t2=0.0
t3=0.2
b1=0.9
b2=2.3
alpha=0
D=1
q=10
```



Esto nos ayudará a identificar los parámetros ingresados. Con la función `typeIWall` habremos creado el muro típico que se muestra en la figura,que habrá generado un modelo con nudos y elementos para el muro. Los parámetros `alpha`, `D` y `q`, corresponden al ángulo de inclinación del terreno, a la profundidad de desplante y a la sobrecarga respectivamente.

- `orient_model!(mywall.model);`

  La entidad creada en el primer paso con la función `typeIwall` tendrá un campo denominado `model`, al cuál se puede ingresar mediante `mywall.model`, este modelo contedrá los nudos y elementos que forman la sección del muro y además, otras propiedades que se explicarán más adelante. Al ejecutar `orient_model!(mywall.model)` se chequeará el modelo, de tal manera que orientará los nudos para cada elemento en el modelo, tal que esten dispuestos en sentido antihorario para la formación del triángulo o cuadrilátero, además, eliminará o corregirá cualquier elemento mal formado o que no forme un elemento triangular o cuadrangular.Por ejemplo en el muro típico que creamos, de ser todos los parámetros diferentes de cero, tendríamos 2 elementos triangulares y 2 elementos cuadrangulares (ver ítem anterior).

- ```
  mywall.D=1;
  mywall.q=10;
  ```

  Aqui, lo que hacemos es establecer nuestra profundidad de desplante (1m) y nuestra sobrecarga 10 Kn/m^2, por defecto el ángulo de inclinación será cero, pero si quisieramos establecerlo lo haríamos de manera similar, estableciendo un valor al campo `mywall.alpha`. Cada vez que querramos ver los valores establecidos para estos parámetros, solo es necesario escribir `mywall`, que es el nombre que le pusimos a nuestro muro.

- ```
  addsoil!(mywall.model,[25 0 16.5]);
  addsoil!(mywall.model,[25 13 16.5]);
  ```

  Aquí agregamos propiedades de suelo, siendo en ese orden el ángulo de fricción, la cohesión y el peso unitario. Puede ingresarse a las propiedades ingresadas mediante `mywall.model.soilprop`, ver más detalles de la función tipeando `?` para entrar al entorno de ayuda de Julia y luego escribiendo el nombre de la función, en este caso `addsoil!`.

- `addmat!(mywall.model,[21000. 0 24]);`

Aquí agregamos propiedades de material, siendo en ese orden la resistencia a la compresión, la resistencia a la tracción y el peso unitario. Puede ingresarse a las propiedades ingresadas mediante `mywall.model.matprop`, al igual que el ítem anterior puede verse más detalles de la función usando el entorno de ayuda de Julia.

- `RetWall.build_rankine_pline(mywall);`

  Para el análisis del muro, es necesario crear una linea de presiones, tanto a la derecha (empuje activo), como a la izquierda (empuje pasivo) del muro, está función llenará los campos `pnod`(nudos), `pliners`(linea de presión a la derecha del muro) y `plinels` (línea de presión a la izquierda del muro) de `mywall.model`.

- `push!(mywall.model.nod,mywall.model.pnod[2:2,:]);`

  Ya que habrá un bloque de suelo que contribuirá a la estabilidad, tendrá que definirse este como parte del muro, 3 puntos de los cuatro que forman este bloque ya están en la matriz de nudos, el 4to nudo está en la segunda entrada de la matriz de nudos de la linea de presiones (pnod) creada con la función anterior.

  Por tanto debe agregarse este último nudo a la matriz de nudos principal, agregar una propiedad del suelo como propiedad de material y agregar el bloque de suelo en la matriz de elementos.

  con la linea `push!(mywall.model.nod,mywall.model.pnod[2:2,:]);` agregamos la segunda entrada de la matriz de nudos de la línea de presiones (`mywall.model.pnod`) a la matriz de nudos principal (`mywall.model.nod`).

- `addmat!(mywall.model,[0 0 16.5]);`

  Agregando suelo como material según lo indicado en el item anterior, solo es necesario ingresar el peso unitario.

- `push!(mywall.model.elm,[3 11 10 8 2]);`

  Agregando el elemento, vinculándolo con el material agregado. Por ser elemento cuadrangular es suficiente agregarlo al final.

- `mywall.model.plinels[1,3]=2;`

  Si se va a usar una propiedad de suelo diferente para la verificación por deslizamiento directamente vinculada con el terreno en el lado izquierdo, debe agregarse

está propiedad con la función `addsoil!` y además vincularla con la línea de presiones a la izquierda.
Con está linea asignamos la segunda propiedad de suelo agregada, a la línea de presiones a la izquierda.

- `mywall.model.soilprop[2,5]=166;`

  Se debe agregar la capacidad portante del terreno. Este valor debera ser ingresado en la quinta columna y en la última fila de la matriz `mywall.model.soilprop`.

  En este caso solo tenemos dos propiedades ingresadas

- `report(mywall,design=1,ignore_pasive_moments=1);`

  Luego de realizar los pasos ya estamos listos para generar el reporte si tenemos LaTeX.

`report(mywall)`, solo realizará el reporte de estabilidad.

En el reporte, en las secciones de verificación del muro, deberá aparecer la expresión **Ok!!** en rojo, si no aparece significa que no está pasando el chequeo, o además, en el caso de la capacidad de carga, puede ser que no se haya ingresado una capacidad portante.

`report(mywall,design=1,ignore_pasive_moments=1);`
Se utiliza la palabra clave `design=1`, para los casos en que el muro de contención requiera diseño de refuerzo(muros en voladizo).
La palabra clave `ignore_pasive_moments=1`, se usa para ignorar los momentos resistentes de la fuerza pasiva.
ambas palabras clave son opcionales.
La primera vez que se ejecute esta acción, nos pedirá que instalemos algunos paquetes de LaTeX, solo aceptamos la instalación.
Otras palabras clave opcionales que pueden usarse con esta función se detallan a continuación:
Palabras clave que solo se verifican cuando la palabra clave para diseño este activa, es decir `design=1`

- `fcv=valor`: factor de carga viva, por defecto se usa `fcv=1.7`.
- `fcm=valor`: factor de carga muerta, por defecto se usa `fcm=1.4`.
- `rp=valor`: recubrimiento en la pantalla, por defecto se usa `rp=0.04`.

- `rz=valor`: recubrimiento en la zapata, por defecto se usa `rz=0.075`.
- `rlist=matriz`: Lista de diámetros y áreas de refuerzo, `matriz` debe ser del tipo `Array{Any,2}`, la primera columna corresponde a la simbología o nombre del refuerzo, la segunda al diámetro (en m) y la tercera al área del refuerzo (en m^2). En cuanto a las filas, las primeras 4 corresponden en ese orden a: refuerzo en la pantalla, refuerzo en la zapata, refuerzo por temperatura en la cara frontal de la pantalla y refuerzo por temperatura en la parte posterior de la pantalla. Ver el segundo ejemplo para ver como crear la matriz.
- `phim=valor`: factor de reducción de resistencia a momentos, por defecto se usa `phim=0.9`.
- `phic=valor`: factor de reducción de resistencia a corte, por defecto se usa `phic =0.85`.
- `fcid=id`: por defecto se usa la resistencia a la compresión del concreto igual a $21000KN/m^2 \approx 210Kg/cm^2$, si se desea utilizar otro valor, debe agregarse como material en `mywall.model.matprop` especificando la resistencia a la compresión (1ra columna). `fcid` será el índice del material ingresado, en otras palabras el número de fila que le corresponde en `mywall.model.matprop`.
- `fyid=id`: por defecto se usa el límite de fluencia del acero igual a $420000KN/m^2 \approx 4200Kg/cm^2$, si se desea utilizar otro valor, debe agregarse como material en `mywall.model.matprop` especificando el límite de fluencia (4ta columna). `fyid` será el índice del material ingresado, en otras palabras el número de fila que le corresponde en `mywall.model.matprop`.

2.2.2. Segundo ejemplo

Como segundo ejemplo, las siguientes líneas muestran el diseño completo de otro muro de contención típico en voladizo, donde se muestra como crear una matriz de refuerzos, para ingresarla como palabra clave para el diseño.

```
mywall=typeIwall(b1=.80,t2=.35,t1=.3,t3=.0,b2=2.85,hz=.7,hp=5.8);
orient_model!(mywall.model);
mywall.D=1;
mywall.alpha=11;
mywall.q=10;
addsoil!(mywall.model,[28 0 18]);
addsoil!(mywall.model,[28 17 18]);
```

```
addmat!(mywall.model,[21000. 0 24]);
RetWall.build_rankine_pline(mywall);
push!(mywall.model.nod,mywall.model.pnod[2:2,:]);
addmat!(mywall.model,[0 0 18]);
addmat!(mywall.model,[0. 0 0 420000]);#para el acero de refuerzo
push!(mywall.model.elm,[3 11 10 8 2]);
mywall.model.plinels[1,3]=2;
mywall.model.soilprop[2,5]=200;
ref=Array{Any,2}([ "\\phi3/4'"  1.905e-2  2.84e-4
        "\\phi5/8'"  0.01588   0.0002
        "\\phi1/2'"  0.0127  1.29e-4
        "\\phi3/8'"  0.00953  7.1e-5
       ]);

report(mywall,design=1,ignore_pasive_moments=1,rlist=ref);
```

2.2.3. Ejemplo usando la teoría de presión de tierra de Coulomb

EL siguiente ejemplo muestra el diseño completo de otro muro de contención típico de gravedad usando la teoría de presión de tierra de Coulomb, los únicos cambios respecto a los ejemplos anteriores es el uso de la función para crear la linea de presiones y la palabra clave que debe incluirse en la función de reporte.

```
mywall=typeIwall(b1=.8,t2=.27,t1=.6,t3=1.53,b2=0.3,hz=.8,hp=5.7);
orient_model!(mywall.model);
mywall.D=1.5;
mywall.alpha=0;
mywall.q=0;
addsoil!(mywall.model,[32 0 18.5]);
addsoil!(mywall.model,[24 30 18]);
addmat!(mywall.model,[21000. 0 23.58]);
RetWall.build_coulomb_pline(mywall);
mywall.model.plinels[1,3]=2;
mywall.model.soilprop[2,5]=200;
report(mywall,ignore_pasive_moments=1,analysis_type="Coulomb");
```

2.2.4. Ejemplo usando la teoría de presión de tierra de Coulomb con condición sísmica

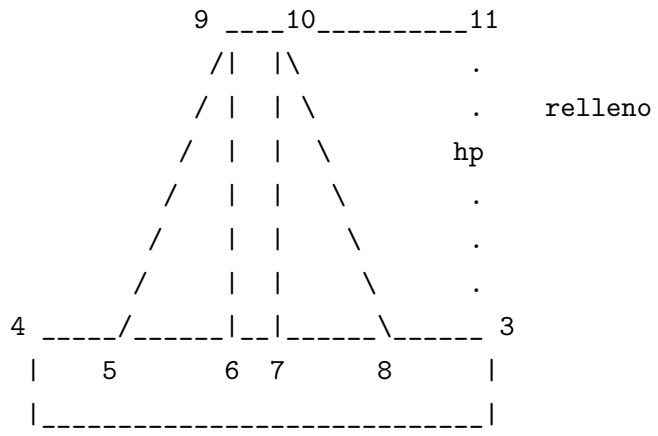
EL siguiente ejemplo muestra el diseño completo de otro muro de contención típico de gravedad usando la teoría de presión de tierra de Coulomb en condición sismica.

```
mywall=typeIwall(b1=.95,t2=.3,t1=.6,t3=1.6,b2=0.3,hz=.8,hp=5.7);
orient_model!(mywall.model);
mywall.D=1.6;
mywall.alpha=0;
mywall.q=0;
addsoil!(mywall.model,[32 0 18.5]);
addsoil!(mywall.model,[24 30 18]);
addmat!(mywall.model,[21000. 0 23]);
RetWall.build_coulomb_pline(mywall);
mywall.model.plinels[1,3]=2;
mywall.model.soilprop[2,5]=200;
RetWall.dynamic_coulomb_report(mywall,kh=0.08944272,kv=0,ignore_pasive_moments=1);
```

2.3. Otros detalles

2.3.1. distribución de nudos y elementos

- Por defecto, el muro generado usando la función `typeIwall`, contendrá un modelo (accesible mediante `mywall.model`) que distribuirá los nudos según como se muestra en la siguiente figura:



```
1                                   2
```

Las coordenadas de estos nudos estarán almacenados en `mywall.model.nod`, tener en cuenta que el nudo número 11, es agregado posteriormente al considerar que habrá un bloque de suelo que contribuirá a la estabilidad, este bloque estará determinado por los nudos 3, 11, 10 y 8 (es por esto el paso donde se usa `push!(mywall.model.elm ,[3 11 10 8 2])` en los ejemplos considerados, siendo el último elemento el índice de la propiedad de suelo en `mywall.model.matprop`)

- La linea de presiones creada con la función `RetWall.build_rankine_pline(mywall);`, está conformada por los puntos 2 y 11 para empuje activo (lado derecho), y el punto 1 y otro que dependerá de la profundidad de desplante, para empuje pasivo (lado izquierdo), las coordenadas de estos nudos están almacenados en `mywall.model .pnod`.
  Ambas lineas de presiones son accesibles mediante `mywall.model.pliners` (línea de empuje activo a la derecha del muro) y `mywall.model.plinels` (línea de empuje pasivo a la izquierda del muro), estos tienen la forma:

```
[2  1  1  0  1]
```

donde las dos primeros números son los índices de los nudos en `mywall.model.pnod`, el tercer número es índice de la propiedad de suelo correspondiente en `mywall.model .soilprop`; el cuarto número corresponde al tipo de presión (0->presión activa,1->presión en reposo y 2->presión pasiva) y el quinto número es la dirección en la que actua el empuje (0-> la presión actua de izquierda a derecha y 1->de derecha a izquierda).

2.4. contacto

Para cualquier duda, comentario, reporte de problemas o sugerencias puede visitarnos a la página de Facebook de LiviCing o al grupo también de LiviCing. También se puede reportar cualquier error en el repositorio de GitHub.