

PRIMER CATHY

Laura Busato

Department of Geosciences - UNIPD

July 8, 2016

Contents

1 Uso del terminale	7
1.1 Fortran	9
1.2 Vim	9
1.3 Gnuplot	11
2 Introduction to CATHY	13
3 Boundary conditions	15
4 Mesh	19
4.1 Creazione della mesh	21
4.2 Mesh 2-D unstructured	23
5 Input file	25
5.1 File di input per il Data Assimilation	32
5.2 File di input non utilizzati	32
6 Compilazione	35
7 Output file	37
7.1 File di output che si possono trascurare	42
8 Data assimilation	45
8.1 Geoelettrica	49
8.2 Eseguibili	50
8.2.1 <code>ert_grid.exe</code>	50

8.2.2 <code>ert_sat3d.exe</code>	54
8.3 Script	55
8.3.1 Eseguibili contenuti nello script	57
8.3.2 Output	57
8.4 Data Assimilation in CATHY	58
8.5 Riassunto dei passaggi da seguire	61
9 Modello piante	63
9.1 Input piante	63
9.2 Output piante	67
10 Rappresentazione degli output: VisIt	69
10.1 Plot di output CATHY	69
10.2 ThreeSlice operator	71
10.3 Isosuperfici	72
Bibliografia	72

List of Figures

3.1	Cubo rappresentante il volume simulato nel caso dell'arancio, con indicato un esempio di BC	15
3.2	Esempio di seepage face con un argine di un corso d'acqua. La curva rossa indica che c'è intersezione tra la tavola d'acqua e il lato esterno dell'argine (seepage face), la curva verde invece non interseca il lato esterno dell'argine e quindi il fenomeno non ha luogo.	16
4.1	Esempi di 2-D structured mesh (mesh strutturata).	19
4.2	Esempio di 3-D structured mesh (mesh strutturata).	20
4.3	Dettaglio su come vengono creati gli elementi tetraedri nella mesh 3-D. Ogni colore rappresenta un diverso elemento tetraedrico. . .	20
5.1	Il pallino rosso rappresenta un nodo e l'area verde è l'"area nodale" che insiste su quel dato nodo, cioè l'area su cui CATHY calcola il flusso che noi riferiamo al singolo nodo.	26
8.1	Data assimilation. La curva nera rappresenta il vero andamento della pressione, la curva azzurra quella simulata da CATHY e le curve colorate e tratteggiate rappresentano diverse simulazioni con diversi parametri e forzanti atmosferiche.	46
8.2	Esempio di zonazione a blocchi.	46
8.3	Grafico andamento degli <i>ensemble</i> durante l'assimilation.	47
10.1	Apertura file con VisIt.	70
10.2	Add dati con VisIt.	71

10.3 Isosuperfici con VisIt.	73
--------------------------------------	----

Chapter 1

Uso del terminale

Il *terminale* è detto anche *shell* ed è necessario utilizzarlo per far girare il modello, in quanto da qui possono essere dati i vari comandi per la compilazione (ma non solo!).

Alcuni comandi più utili da terminale:

cd /percorso/cartella entra nella cartella specificata;

cp file_da_copiare /home/utente/cartella_in_cui_copiare copia il file indicato nella cartella specificata;

cp -r sottocartella /home/utente/cartella_in_cui_copiare copia la sottocartella nella cartella specificata;

dpkg --get-selections >/pacchetti_installati.txt salva la lista dei pacchetti installati, se la si salva può essere usata per ripristinare il sistema;

echo scrive ogni stringa data nello standard output, con uno spazio tra l'una e l'altra e va a capo dopo l'ultima;

free mostra lo stato della memoria;

free -m mostrato lo stato della memoria in Mb;

gedit nomefile apre il file con gedit (editor di testo);

grep parola *.estensione cerca la parola “parola” in tutti i file con estensione “estensione”. Ad esempio **grep 660 *.f** cerca la parola “660” in tutti i file con estensione .f;

history stampa la lista degli ultimi comandi lanciati da terminale;

ln -s percorso file crea un link al file indicato nella cartella in cui ci si trova;

locate file1 trova tutte le occorrenze di “file1”;

ls stampa il contenuto della cartella;

ls -a stampa il contenuto della cartella mostrando anche i contenuti nascosti;

ls -l stampa il contenuto della cartella in formato elencando tutti i dettagli dei file/sottocartelle contenuti;

mkdir /home/utente/nuova_cartella crea una nuova cartella;

mv file_da_spostare /home/utente/cartella_in_cui_copiare sposta il file indicato (o una cartella) nella cartella specificata;

mv nome_vecchio nome_nuovo rinomina un file;

open nomefile apre il file “nomefile” selezionando autonomamente il programma predefinito per la sua apertura;

pdftk file_uno.pdf file_due.pdf file_tre.pdf cat output123.pdf unisce più pdf in un unico file;

ps2pdf nomefile converte il file da Postscript a .pdf;

pwd mostra il percorso della directory di lavoro corrente;

rm -rf /home/utente/cartella_da_eliminare elimina la cartella e gli eventuali file al suo interno;

rmdir /home/utente/cartella_da_eliminare per eliminare una cartella vuota;

sudo apt-get install nome_pacchetto installa un nuovo pacchetto;

```
sudo apt-get remove nome_pacchetto rimuove il pacchetto selezionato;  
  
sudo halt spegne il computer ora;  
  
sudo poweroff spegne il computer ora;  
  
sudo reboot riavvia il computer da terminale;  
  
sudo shutdown -h now spegne il computer ora;  
  
sudo shutdown -h +30 spegne il computer tra 30 minuti;  
  
sudo shutdown -h 3107150830 spegne il computer il 31 luglio 2015 alle 08.30;  
  
top mostra i processi in esecuzione;
```

- ◊ : non usare mai **rm** per cancellare qualcosa. È sempre preferibile cancellare dalle cartelle o, al massimo, usare **rmi**.
- ◊ : il tasto “TAB” autocompleta il nome del file o della cartella.
- ◊ : per aprire un file da terminale e lasciare il terminale attivo, alla fine dei comandi **open** e **gedit** aggiungere il simbolo “&”.

Per maggiori dettagli guardare il file “Ubuntu_comandi_da_terminale.pdf”.

1.1 Fortran

Una volta creato il file sorgente **nomefile.f**, per compilarlo e creare l'eseguibile si dà da terminale il seguente comando: **gfortran nomefile.f -o nomefile.exe**. Posso scegliere io quale nome dare all'eseguibile.

- ◊ : nel caso dell'eseguibile di CATHY devo usare il comando **make** (vedi cap. 6)

1.2 Vim

Vim è un editor di testo open source e multipiattaforma. Per aprire un file in vim è sufficiente dare il comando da terminale **vim nomefile**, facendo attenzione a trovarsi nella cartella contenente quel dato file. Se il file non è presente,

verrà creato al primo salvataggio di vim. All'apertura del documento vim è in "command mode". Se si vuole editarlo è necessario passare all'"insert mode" premendo il tasto **i**. A questo punto possiamo editare il file come in un normale editor testuale, con la differenza che non potremo usare il mouse. Per muovere il cursore all'interno del documento in "insert mode" possiamo usare le frecce. Una volta modificato il documento, per salvarlo dobbiamo tornare in "command mode": per fare questo premiamo il tasto **esc**. Tutti i comandi di vim, ad eccezione dei comandi di navigazione all'interno del documento, sono preceduti da due punti (:). Il comando viene scritto nella parte bassa del terminale e viene eseguito quando si preme invio.

Elenco sintetico dei comandi e dei tasti più utili:

G va alla fine del file;

142G va alla riga 142, contata a partire dalla prima riga del file;

gg va all'inizio del file;

n continua la ricerca in giù;

N continua la ricerca in su;

/testo cerca "testo" dalla posizione attuale in giù;

?testo cerca "testo" dalla posizione attuale in su;

p incolla la riga in memoria;

:q per uscire;

:q! per uscire senza salvare;

:w per salvare;

:wq per salvare e uscire;

yy copia la riga attuale (ad es., 20yy copia 20 righe);

:e.. apre in vim la cartella del file in cui sto lavorando.

1.3 Gnuplot

Gnuplot is a portable command-line driven graphing utility. È possibile utilizzarlo non solo per creare grafici, ma anche per effettuare calcoli. Per attivarlo, è sufficiente digitare **gnuplot** a terminale, in questo modo sarà attivo nella cartella in cui ci trova. Se, però, si volesse cambiare cartella è sufficiente utilizzare il comando indicato sotto. Quando si lavora in gnuplot è presente il prompt “**gnuplot>**”. Lista di alcuni comandi*:

!cd ‘percorso’ cambio cartella;

f(x)=ax2+b*x+c** consente di creare la propria funzione (in questo caso, una parabola). È possibile definire i valori dei parametri con il comando **a=1.;b=3.;c=2.** (il “;” è l’equivalente dell’andata a capo). È possibile plottare la funzione con **plot f(x)** e calcolare il valore della funzione in un certo punto con **pr f(5.2)**;

plot ‘file_dati.txt’ using 1:4 plotta i dati contenuti nel file “file_dati.txt” usando la colonna 1 come ascissa e la colonna 4 come ordinata (le colonne sono generalmente separate da spazi o tab, il carattere # indica una riga di commento e non viene considerata). Nel caso in cui non si specifichino le colonne da considerare, vengono usate di default la colonna 1 per le ascisse e la colonna 2 per le ordinate;

plot ‘file_dati.txt’ t‘colonna 2’ attribuisce l’etichetta “colonna 2” ai dati;

plot [xmin:xmax][ymin:ymax] sin(x) crea il grafico della funzione $\sin(x)$ nell’intervallo indicato, pur calcolando lo stesso i valori omessi;

plot sin(x) crea il grafico della funzione $\sin(x)$;

plot sin(x), cos(x) crea i grafici delle due funzioni $\sin(x)$ e $\cos(x)$ nella stessa finestra;

*Una lista di comandi più completa è disponibile al link http://webusers.fis.uniroma3.it/meneghini/LPC/files_lezioni/Guida_gnuplot.htm

pr 3.5*sqrt(2) esegue il calcolo dell'espressione e ne stampa il risultato. NB:

se si utilizzano solo numeri interi, il risultato sarà anch'esso intero, omettendo così l'eventuale parte decimale presente. È sufficiente scrivere un solo numero in forma reale (ad es., 2.0) per ottenere un risultato in forma reale;

set autoscale determinazione automatica dell'intervallo;

set key top left riposiziona la legenda in alto a sinistra (altre opzioni: top/bottom, right/left);

set title "titolo" titolo del grafico;

set xlabel "asse x" etichetta asse x ;

set xrange [valore:valore] intervallo x

set ylabel "asse y" etichetta asse y ;

set yrange [valore:valore] intervallo y ;

splot sin(x*y) crea il grafico 3D della funzione $\sin(x * y)$;

test apre una finestra grafica e mostra un test delle possibilità grafiche;

unset key disabilita la legenda;

Chapter 2

Introduction to CATHY

The CATchment HYdrology (CATHY) model couples a finite element solver for the Richards equation describing flow in variably saturated porous media and a finite difference solver for the diffusion wave equation describing surface flow propagation throughout a hill slope and stream channel network identified using terrain topography and the hydraulic geometry concept [1].

The mathematical model is described by a *system of two partial differential equations* [1]:

1. Subsurface flow equation:

$$S_w S_s \frac{\partial \psi}{\partial t} + \phi \frac{\partial S_w}{\partial t} = \nabla \cdot [K_s K_r (\nabla \psi + \eta_z)] + q_{ss} \quad (2.1)$$

2. Surface flow equation:

$$\frac{\partial Q}{\partial t} + c_k \frac{\partial Q}{\partial s} = D_h \frac{\partial^2 Q}{\partial s^2} + c_k q_s \quad (2.2)$$

In (2.1): $S_w = \theta/\theta_s$ is water saturation (grado di saturazione in acqua) [-], θ is the volumetric moisture content (contenuto idrico volumetrico) [-], θ_s is the saturated moisture content (generally equal to the porosity ϕ), S_s is the aquifer specific storage coefficient [L^{-1}], ψ is pressure head [L] (carico di pressione), t is time [T], ∇ is the gradient operator [L^{-1}], K_s is the saturated hydraulic conductivity tensor [L/T], $K_r(\psi)$ is the relative hydraulic conductivity function

$[-]$, $\eta_z = (0, 0, 1)'$, z is the vertical coordinate directed upward $[L]$, and q_{ss} represents distributed source (positive) or sink (negative) terms $[L^3/L^3T]$ [1].

In (2.2) a 1-D coordinate system s $[L]$ is used to describe each element of the channel network. In this equation, Q is the discharge along the rivulet/stream channel $[L^3/T]$, c_k is the kinematic celerity $[L/T]$, D_h is the hydraulic diffusivity $[L^2/T]$, and q_s is the inflow (positive) or outflow (negative) rate from the subsurface to the surface $[L^3/LT]$ [1].

The 3-D Richards equation is solved numerically by Galerkin finite elements in space using tetrahedral elements and linear basis functions, and by a weighted finite difference scheme for integration in time. The nonlinear characteristic relationships $K_r(\psi)$ and $S_w(\psi)$ are specified using either *van Genuchten and Nielsen* [1985], *Brooks and Corey* [1964], or *Huyakorn et al.* [1984] expression. Linearization via Newton or Picard iteration is used in the solution procedure [1].

Cioè, uso van Genuchten per descrivere la dipendenza di S_w e K_r dalla pressione: così posso definire su tutto il dominio, in modo uniforme, le curve di resistività capillare. In alternativa, posso usare le cosiddette “look-up table”, cioè delle curve definite per punti, utili nel caso in cui siano presenti delle eterogeneità.

◇ : nel caso dell’arancio non utilizzo il modulo superficiale in quanto la parte di flusso che non si infiltra non viene considerata. More details on the numerical aspects and other features of the subsurface solver can be found in the work by *Paniconi and Putti* [1994].

Chapter 3

Boundary conditions

◊ : tutte le unità utilizzate in ogni file di CATHY devono essere tra di loro consistenti. Ad es.: m, m/s, s (in genere si lavora in secondi). In altre parole, se come unità di misura della lunghezza uso i metri, tutte le lunghezze devono essere espresse in metri, e così via.

We have two types of boundary conditions (BC): Neumann BC (or specified flux) and Dirichlet BC (or pressure).

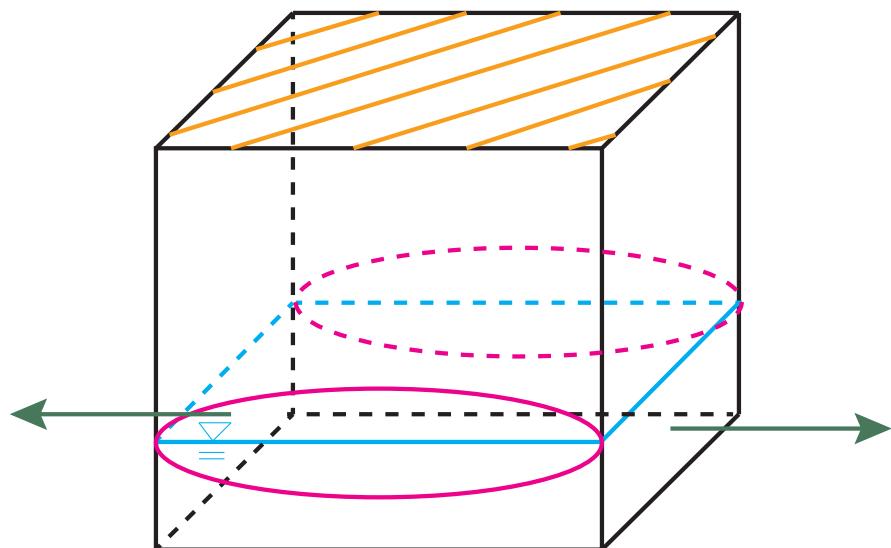


Fig. 3.1: Cubo rappresentante il volume simulato nel caso dell'arancio, con indicato un esempio di BC.

Con riferimento al cubo in Fig. 3.1:

- Faccia superiore (arancione): condizioni atmosferiche che interagiscono con

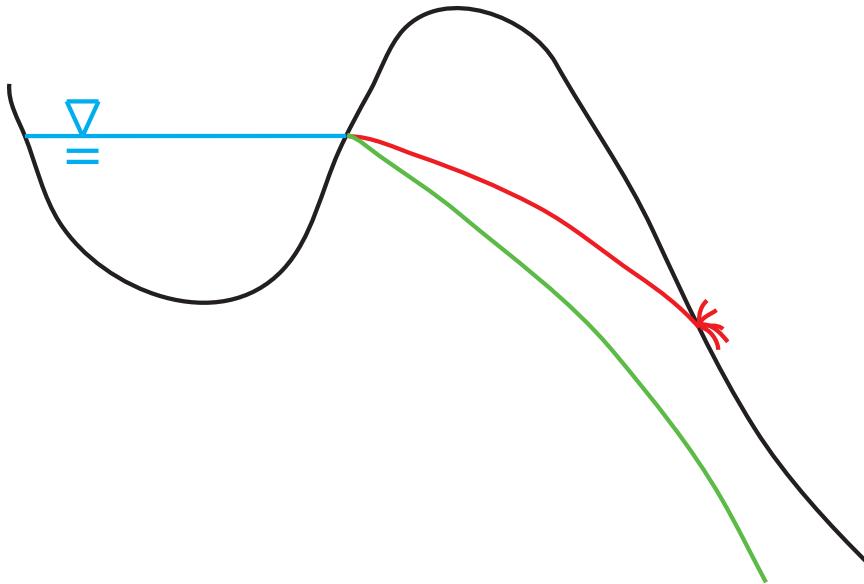


Fig. 3.2: Esempio di seepage face con un argine di un corso d'acqua. La curva rossa indica che c'è intersezione tra la tavola d'acqua e il lato esterno dell'argine (seepage face), la curva verde invece non interseca il lato esterno dell'argine e quindi il fenomeno non ha luogo.

le piante. Pioggia: condizioni di Neumann. Quando non ci può più essere infiltrazione metto Dirichlet. Evaporazione: si indica un limite di pressione minimo (P_{min}) al di sotto del quale si ha uno switch da Neumann a Dirichlet (in quanto al di sotto di questo valore non si ha più evapotraspirazione). Posso mettere delle BC uniformi su tutta la superficie o specificarle nodo per nodo.

The boundary condition for any given surface node can switch between a Dirichlet condition and a Neumann condition depending on the saturation (or pressure) state of that node. A Neumann (or specified flux) boundary condition corresponds to atmosphere-controlled infiltration or exfiltration, with the flux equal to the rainfall or potential evaporation rate given by the atmospheric input data. When the surface node reaches a threshold level of saturation or moisture deficit, the boundary condition is switched to a Dirichlet (specified head) condition, and the infiltration or exfiltration process becomes soil limited [1].

\diamond : in case of simultaneous precipitation and evaporation, we impose at

the surface the net flux, i.e., precipitation minus evaporation.

- Pareti e bottom: come prima opzione posso mettere no-flow sulle pareti e scegliere Neumann o Dirichlet sul bottom. In alternativa, posso fare con in Fig. 3.1, cioè mettere su due facce il no-flow e sulle altre due condizioni di Dirichlet. In questo modo ho che la tavola d'acqua varia come le curve rosse in figura dove ho no-flow, mentre ho flusso in uscita (frecce verdi) o in entrata sulle facce dove ho imposto Dirichlet.

◊ : dove non metto nulla ho condizioni di no-flow, altrimenti le devo specificare.
- Seepage face: Quando l'acqua interseca l'argine ha luogo il deflusso (linea rossa in Fig. 3.2). Così ho a sinistra una pressione positiva, mentre a destra Dirichlet è pari a 0. Quando la pressione a sinistra diventa negativa (come nel caso della linea verde), allora la condizione a destra diventa di no-flow. Sono però io a dover indicare quali sono i nodi influenzati.

Chapter 4

Mesh

◊ : tutte le unità utilizzate in ogni file di CATHY devono essere tra di loro consistenti. Ad es.: m, m/s, s (in genere si lavora in secondi). In altre parole, se come unità di misura della lunghezza uso i metri, tutte le lunghezze devono essere espresse in metri, e così via.

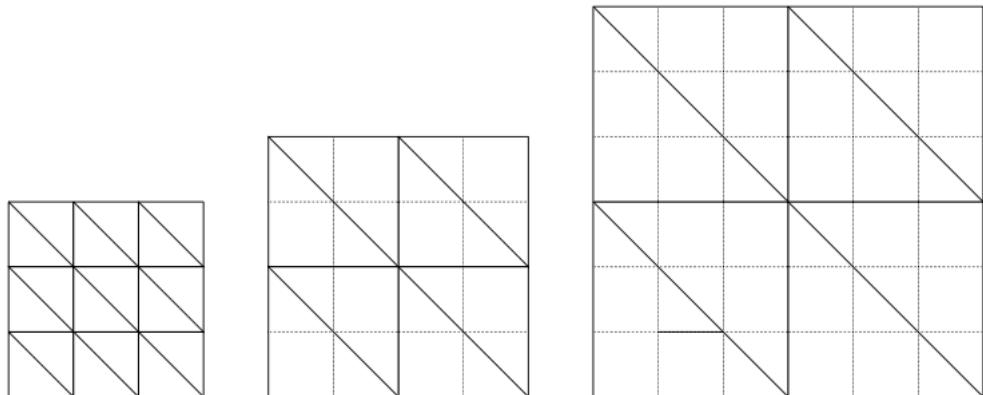


Fig. 4.1: Esempi di 2-D structured mesh (mesh strutturata).

Due to the 1-D nature of the surface flow module versus the 3-D nature of the subsurface module, and the cell-based spatial discretization of the former with respect to the node-based discretization of the latter, some issues arise when exchanging information between the two modules [1].

The cell discretization produces two different numberings: a complete one for the subsurface grid, including pit cells, and a partial one for surface discretization, without pit cells. Each cell of the complete surface grid is subdivided into two

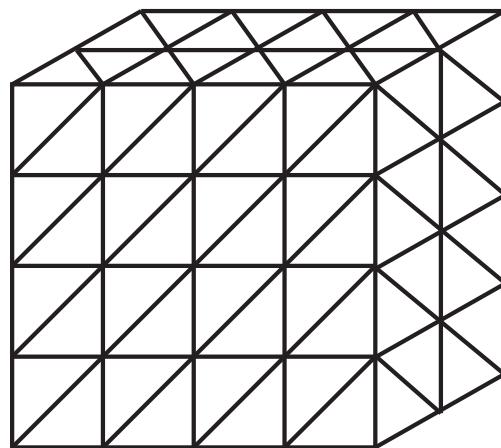


Fig. 4.2: Esempio di 3-D structured mesh (mesh strutturata).

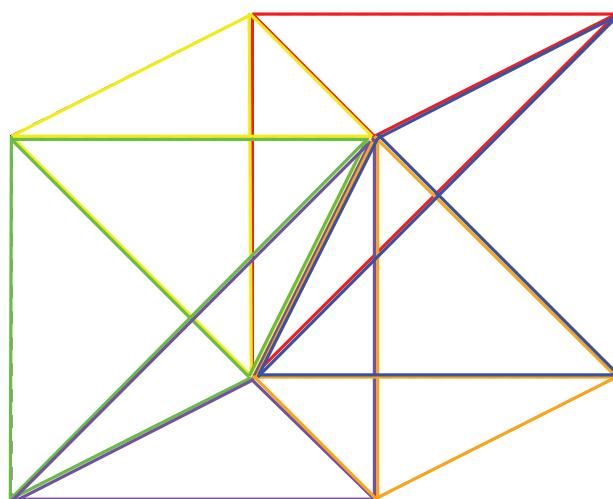


Fig. 4.3: Dettaglio su come vengono creati gli elementi tetraedri nella mesh 3-D. Ogni colore rappresenta un diverso elemento tetraedrico.

triangles (Fig. 4.1), which are in turn projected along the vertical to create the *3-D tetrahedral structured finite element mesh* [1].

Io creo una mesh qualunque superficiale, poi il codice la replica automaticamente in profondità. Si creano così degli STRATI in profondità, i quali non sono però definiti in senso fisico, bensì intesi come strati di griglia. Per ognuno devo specificare i vari parametri ed essi posso anche essere uguali per i diversi strati. Nel caso in cui io avessi eterogeneità, devo prima crearle in superficie. Genero così delle ZONE a cui attribuisco un parametro. Così posso definire le varie parti della mia mesh attribuendo i parametri (K_s , porosità e S_s) in base a ZONE e STRATI.

◊ : la mesh deve essere strutturata! Un esempio è quello in Fig. 4.2, mentre la figura Fig. 4.3 mostra come vengono costruiti i singoli elementi tetraedri.

4.1 Creazione della mesh

La creazione della mesh strutturata 3-D si effettua in due passaggi distinti: prima si crea la mesh superficiale, che deve tenere conto delle eterogeneità presenti nel sottosuolo, e poi la si proietta lungo z .

1. Per creare la mesh superficiale si utilizza il contenuto della cartella `co dici→gridgen_2d`. Per prima cosa si inseriscono nel file `mesh.data` le info richieste, ovvero:

- `length_x`: lunghezza mesh lungo x ;
- `length_y`: lunghezza mesh lungo y ;
- `ncelx`: quante celle (elementi) voglio lungo l'asse x ;
- `ncely`: quante celle (elementi) voglio lungo l'asse y .

Quindi si fa girare l'eseguibile `mesh.exe` dando il comando `./mesh.exe` da terminale. Terminato questo primo passaggio, si fa girare l'eseguibile `coor1.exe`.

◇ : nel caso in cui siano stati modificati i file `mesh.f` e `coor1.f` è necessario ri-compilare gli eseguibili con il comando in sezione 1.1 ed utilizzarli al posto di quelli vecchi.

Il questo modo si ottengono diversi file di output:

- `coord1.coord`: contiene le coordinate di tutti i nodi nella forma $(x; y)$;
 - `coor1.out`: contiene un riassunto di tutte le informazioni riguardanti la mesh superficiale. Qui vedo il numero di nodi e di elementi;
 - `mesh1.triang`: ogni riga di questo file è riferita ad un triangolo della mesh superficiale. Le prime tre colonne indicano i nodi che costituiscono quel dato triangolo, mentre l'ultima colonna indica la zona a cui quel triangolo appartiene. I nodi vengono letti in senso antiorario.
2. Ora mi sposto nel file `grid` contenuto nella cartella `input` di ciascuna simulazione. Questo file consente, una volta che viene fatto girare il modello, di creare la mesh 3-D proiettando lungo z la mesh superficiale. In questo file devo inserire:

- NZONE: numero di tipi di materiali che costituiscono il mezzo poroso;
- NSTR: numero di strati di griglia lungo z ;
- N1: resta uguale a 30;
- NNOD: numero di nodi (lo si ricava dal file `coor1.out`);
- NTRI: numero di elementi triangolari (lo si ricava dal file `coor1.out`);
- IVERT: indica se gli strati sono orizzontali o meno;
- ISP: indica se la superficie è piatta o meno;
- BASE: profondità della mesh lungo z ;
- ZRATIO: mi indica lo spessore di ciascun strato espresso come %.
Ad esempio, se NSTR=4, allora ZRATIO deve avere 4 valori, la cui somma deve essere pari a 1.0, come $0.25+0.25+0.20+0.30=1.0$.

- Infine inserisco tutti gli elementi della mesh superficiale (le quattro colonne dal file `mesh1.triang`) e poi le coordinate di tutti i nodi (prese dal file `coor1.coord`);
- $Z(1)$: indica l'altezza della griglia superficiale (ad es., se il piano campana è a 0 m, o -1 m, ecc).

4.2 Mesh 2-D unstructured

È possibile utilizzare anche mesh 2-D non strutturate, le quali consentono di variare la dimensione degli elementi a seconda delle necessità del modello (i.e., elementi più piccoli dove voglio una maggiore risoluzione e viceversa). Questa tipologia di mesh può essere creata con software quali MATLAB o GMSH: è sufficiente esportare gli elementi ed i nodi ed inserirli nel file `grid`, come indicato precedentemente. Come già descritto, sarà poi CATHY a creare la mesh 3-D estrudendo la superficie 2-D lungo z .

Chapter 5

Input file

I file di input sono contenuti nella cartella `input`, situata all'interno della cartella di simulazione (vedi capitolo 6). Gli eventuali file che non vengono utilizzati devono comunque essere lasciati all'interno della cartella.

Nel caso dell'arancio i file utilizzati sono:

- `atmbc` (Atmospheric Boundary Conditions):
 - HSPATM:
 - =0 for spatially variable atmospheric boundary conditions input;
 - =blank or 9999 if unit IIN6 (/input/atmbc, questo percorso lo vedo nel file `cathy.fnames`) input is to be ignored;
 - otherwise atmospheric BCs are homogeneous in space;
 - IETO:
 - =0 for linear interpolation of the atmospheric boundary condition inputs between different ATMTIM; otherwise, the inputs are assigned as a piecewise constant function (ietograph);
 - **Quale valore di Q va inserito?**

Q deve essere espresso in [L/T] (ad esempio, m/s se si sta lavorando con queste unità di misura in tutto il modello – le unità di misura devono essere coerenti in tutti i file!). Nel caso delle `atmbc` (forse), CATHY vuole il valore riferito “al nodo” e poi lo plotta su tutta l’area calcolata a congiungendo i centroidi degli

elementi 2-D che insistono su quel dato nodo (area verde in Fig. 5.1). Quindi, se il flusso insiste su più nodi, esso dovrà essere suddiviso nei vari nodi e diviso per l'area che soggiace a ciascun nodo (NB: se la mesh è omogenea, allora l'area sarà la stessa per ogni nodo!).

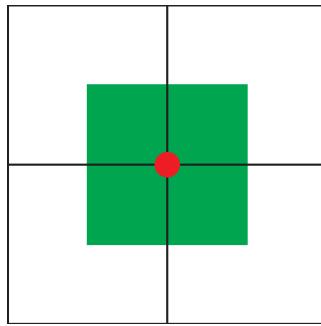


Fig. 5.1: Il pallino rosso rappresenta un nodo e l'area verde è l’“area nodale” che insiste su quel dato nodo, cioè l’area su cui CATHY calcola il flusso che noi riferiamo al singolo nodo.

◊: ATTENZIONE! Quando si inserisce un flusso come atmospheric B.C., la quantità di acqua che effettivamente si infiltrà dipende dalla conducibilità idraulica del suolo. Ad esempio, se immetto 4 l/h ma $K=1$ l/h, allora i 3 litri che non entrano vengono “persi”, cioè trasformati in runoff superficiale.

- **grid**: già illustrato nella sezione 4.1
- **ic** (Initial Conditions):
 - INDP: flag for pressure head initial conditions (se questo è = 3 allora CATHY usa la subroutine ICVHWT). Nel caso in cui si conoscano i valori di pressure head in ciascun nodo (ad es., da una simulazione precedente) e si vogliano utilizzare questi valori come initial condition, allora INDP=1 e, di seguito, si inserisce il corrispettivo valore di pressure head per ciascun nodo;
 - IPOND: flag for ponding head initial conditions;

- WTPOSITION: flag per la posizione della tavola d'acqua rispetto alla base della mesh 3-D. I valori lungo la dimensione della profondità al di sotto del piano campagna sono negativi. Quindi se WTPOSITION è *negativo* sto posizionando la tavola d'acqua al di sotto della base della mia mesh (e viceversa, se questo parametro è positivo, allora la tavola d'acqua è al di sopra della base della mesh).
- **nansfdircbc** (Non-Atmospheric Non Seepage Face Dirichlet Boundary Conditions):
 - TIME: time at current time level;
 - NDIR: number of non-atmospheric, non seepage face Dirichlet nodes in 2-D mesh. Questi nodi vengono replicati verticalmente (ad esempio, se voglio attribuire delle BC a tutta una faccia laterale di un cubo, è sufficiente che io definisca le BC solo sui nodi nel vertice superiore di questa faccia, in quanto poi questi nodi saranno replicati fino alla base della mesh);
 - NDIRC: number of “fixed” non-atmospheric, non seepage face Dirichlet nodes in 3-D mesh (“fixed” means that these BC are not replicated to other nodes - compare NDIR). Nel file **nansfdircbc** utilizzato nella Prova_1 ho:

0.0	TIME
0	0 NDIR NDIRC
1e40	TIME
0	0 NDIR

Tab. 5.1: Esempio di file nansfcirbc tratto da Prova_1.

Questo significa che al time 0.0 non ho nodi con Dirichlet BC e questa condizione vale fino al tempo 1e40 s (cioè all'infinito). Se avessi delle variazioni dovrei indicare il nodo ed il valore di pressione:

◇: Ricorda di modificare il valore di NPMAX nel file **CATHY.H** nel caso in cui

```

time 0      n
numero nodi da 1 a n
pressioni   da 1 a n

```

Tab. 5.2: Eventuali variazioni nel tempo nel file `nansfdircbc`.

si inseriscano dei NDIRC ed il valore di NP2MAX nel caso si inseriscano dei NDIR. I valori di NMAX e NP2MAX corrispondono al numero massimo di nodi NDIRC e NDIR che si possono inserire.

- `nansfneubc` (Non-Atmospherics Non Seepage Face Neumann Boundary Conditions):
 - TIME;
 - NQ: number of non-atmospheric, non seepage face Neumann nodes in 3-D mesh. In realtà qui vi sono due valori (0 e NQ): il primo resta invariato (anche se non so perché), mentre di volta in volta modifico NQ con il numero di nodi interessati dal flusso;
 - CONTQ: non-atmospheric, non-seepage face Neumann nodes numbers in 3-D mesh. Ci devono essere listati NQ valori;
 - Q: non-atmospheric, non-seepage face Neumann values at current time level. Esempio:

0.000		TIME
0	89	NQ
<i>NQ=89volte</i>		
1445 1446 ... 2037		CONTQ
<i>NQ=89volte</i>		
1.6531e - 07 1.6531e - 07 ... 1.6531e - 07		Q
10.000		TIME
0	89	NQ
<i>NQ=89volte</i>		
1445 1446 ... 2037		CONTQ
<i>NQ=89volte</i>		
1.2477e - 07 1.2477e - 07 ... 1.2477e - 07		Q

· **Quale valore di Q va inserito?**

Il questo caso, a differenza di quanto visto per le `atmbc`, Q deve essere espresso in $[L^3/T]$ (ad esempio, m^3/s se si sta lavorando con queste unità di misura in tutto il modello – le unità di misura devono essere coerenti in tutti i file!).

– `parm` (Parameters). Io posso modificare:

- IPRT1: flag for output of input and coordinate data in subroutines `datin.f` and `gen3D.f`. Tra i vari valori che posso attribuirgli, scelgo “3” per ottenere informazioni sulla mesh 3-D. In particolare, dopo aver creato la mesh 2-D ed aver messo `IPRT1=3`, faccio girare CATHY. In output guardo il file `xyz`, `grid2d.exp` e `grid3d` (vedi cap.7);
- DAFLAG: flag for the choice of the data assimilation scheme. Il valore verrà scelto in base al tipo di Data Assimilation;
- ISIMGR: flag for type of simulation and type of surface grid;
- DELTAT: initial and current FLOW3D time step size ($\geq 1.0e + 15$ on input indicates a steady state problem). È il passo di time step iniziale;
- DTMIN: minimum FLOW3D time step size allowed. Al di sotto di questo parametro si blocca;
- DTMAX: maximum FLOW3D time step size allowed. È un upper boundary per il time step. Se questo valore viene superato, la simulazione non si blocca, ma potrei non riuscire a leggere tutti gli input. Se ho input ogni ora, devo avere un time step al massimo ogni mezz'ora - 20 minuti: in altre parole, il time step deve essere un sottomultiplo del “tempo” dei miei input. Infatti, se avessi un time step di 45 minuti, non riuscirei a leggere né il secondo input (dopo un'ora), né il terzo (dopo 2 ore), ma solo il quarto (dopo 3 ore);
- TMAX: time at end of simulation (is set to 0.0 for steady state problems);

- IPRT: flag for detailed output at all nodes and velocity and water saturation (or content?) output at all elements. Governa cosa viene stampato dei DETAILED OUTPUT.
- VTKF: flag for the output in the vtkf files at the detailed output times;
- NPRT: number of time values for detailed nodal output and element velocity output. Numero di output dettagliati che voglio (see description of IPRT and TIMPRT);
- TIMPRT: time values for detailed output. Detailed output is produced at initial conditions (TIME=0), at time values indicated in TIMPRT, and at the end of the simulation (TIME=TMAX). I DETAILED OUTPUT vengono stampati solo ai time step che mi interessano: infatti, non posso stampare un output per tutta la griglia ad ogni time step, altrimenti avrei Giga di file. Devo scegliere io quali time step voglio vedere;

DETAILED OUTPUT consists of: values of pressure head, velocity, water saturation, and relative conductivity (depending on setting of IPRT) at all nodes, velocity and water saturation (depending on setting of IPRT) at all elements; vertical profiles of pressure head, water saturation, and relative conductivity for the NODVP surface nodes, pressure head, water saturation, and SATSUR values at the surface nodes; Nel file `parm` della Prova_1 si ha che i numeri nella riga 14 (2 2 18) corrispondono rispettivamente a IPRT, VTKF e NPRT. I valori da 480 a 182040 sono invece i valori di TIMPRT. Infatti TIMPRT(I), I=1,NPRT vuol dire che devo mettere i valori di TIMPRT per I che va da 1 a NPRT;

- NUMVP: number of surface nodes for vertical profile output. Crea il profilo su quella colonna di nodi per ogni time step dettagliato. L'output si chiama `vp`;
- NODVP: node number for surface nodes selected for vertical profile

output. Metto prima NUMVP (ad es., 2) e poi il numero di nodi che mi interessano;

- NR: number of nodes selected for partial output. Nodi a cui stampa l'output ad ogni iterazione (in genere solo il valore di pressione). Anche qui prima metto il numero di nodi e poi l'elenco.

◇ : per evitare il back-stepping (cioè la capacità del CATHY di modificare DELTAT) è sufficiente porre DTMIN=DTMAX=DELTAT. Attenzione però: così facendo i tempi di simulazione potrebbero aumentare e nel caso di necessità di back stepping la simulazione si bloccherebbe.

– soil:

- PMIN: “air dry” pressure head value (for switching control of atmospheric boundary conditions during evaporation). Pressione minima. In caso di evaporazione, le condizioni superficiali sono di Neumann e la pressione continua a decrescere. Quando la pressione raggiunge il valore di PMIN, le condizioni al contorno passano da Neumann a Dirichlet, in modo che le pressioni non possano scendere al di sotto di PMIN. È una specie di “valore minimo” che si assegna alle pressioni in superficie.
=-9999= libertà alla pressione di diminuire.
= valore di pressione che corrisponde alla saturazione minima raggiunta dal suolo.
- IVGHU: indica come vengono calcolate le curve di risalita capillare, cioè se si utilizza Van Genuchten o le look-up tables (o una delle altre possibilità);
- VGN, VGM, VGRMC, VGPSAT: parametri per le curve di van Genuchten (eq. (5.1)):
 - VGN: n ;
 - VGRMC: residual moisture content θ_r ;
 - VGPSAT: è pari a $-1/\alpha$ (con α espresso in $[L^{-1}]$);

Gli altri parametri sono per altri tipi di moisture curves.

- PERMX (NSTR, NZONE): saturated hydraulic conductivity - xx ;
- PERMY (NSTR, NZONE): saturated hydraulic conductivity - yy ;
- PERMZ (NSTR, NZONE): saturated hydraulic conductivity - zz ;
- ELSTOR (NSTR, NZONE): specific storage;
- POROS (NSTR, NZONE): porosity (moisture content at saturation).

The soil-water content as a function of the pressure head is given by (van Genuchten equation):

$$\theta = \theta_r + \frac{(\theta_s - \theta_r)}{[1 + (\alpha h)^n]^m} \quad (5.1)$$

dove: θ_r è il residual water content, θ_s è il saturated water content (uguale alla porosità!), h è il pressure head e α , m e n sono dei parametri da stimare. In particolare, $m = 1 - (1/n)$.

5.1 File di input per il Data Assimilation

Per effettuare il DA è necessario settare alcuni file di input di CATHY:

- **archie**: si veda la sezione 8.4;
- **data_ass**: si veda la sezione 8.4;
- **elec_nodes**:
 - **N_EL**: numero di elettrodi;
 - **NNOD**: numero del nodo su cui cade l'elettrodo specificato da ELEC;

5.2 File di input non utilizzati

I seguenti file non vengono utilizzati, ma devono comunque essere presenti nella cartella **input**:

- **base_map**;

- `dem_parameters`;
- `depit`;
- `effraininp`;
- `livelli_iniz_s`;
- `mesh`;
- `nudging`;
- `posizione_serb`;
- `retctab`;
- `sfbc`;

Chapter 6

Compilazione

Prima di tutto è necessario creare il file eseguibile `cathy.exe`. Per fare questo, entro nella cartella `codici→code_cathy` e, da terminale, do i comandi `make clean` e `make` all'interno di questa cartella. La cartella `code_cathy` contiene:

- il `Makefile`, all'interno del quale sono contenuti il compilatore, le librerie, gli oggetti ed il nome dell'eseguibile che verrà creato (è qui che, se voglio, posso modificarlo!);
- il file `cathy.fnames`, che serve per indicare dove sono i file di input ed output. Ad esempio, scrivere in “unit IIN1” o “10” è la stessa cosa;
- le seguenti subroutine:
 - `cathy_main.f`: nella parte iniziale contiene la storia del codice, poi il manuale del codice, dove sono spiegate le variabili utilizzate nei file (nelle cartelle) di `input` e `output`;
 - `datin.f`: riguarda gli input e output. Per sapere come un file viene letto lo leggo qui e cerco il nome del file che mi serve così come è scritto nel file `cathy.fnames`. Ad esempio, il file `grid` (contente le info sulla mesh) viene indicato anche come “unit IIN2” o “label 11” (NB: nel `datin.f` è sufficiente cercare “/IIN2”). In questo file vedo anche come un file viene scritto;

- **CATHY.H**: contiene i dimensionamenti massimi, cioè quanta memoria viene data a ciascuna variabile. Ricordati di cambiarli se necessario (ad esempio, MAXSTR = 20, cioè il numero massimo di strati nella mesh 3-D è pari a 20).

◊ : ogni volta che modifco qualcosa nei file sorgente dell'eseguibile (ad es., il numero massimo di nodi) devo ricompilarlo (cioè rifare la procedura **make clean** + **make**)

Mi sposto quindi all'interno della cartella **simulazioni**, dove trovo tutte le simulazioni fatte suddivise per data (oltre a quelle di prova). A ciascuna cartella corrisponde una simulazione, e quindi, per far girare il modello, deve contenere:

- il file eseguibile (ad es., **cathy.exe**) appena ricompilato;
- il file **cathy.fnames**;
- la directory **input**, che contiene tutti i file di input (ad es., **parm**);
- la directory **output**, che contiene i file di output. È necessaria la sua presenza, ma non è necessario creare i file al suo interno;
- la directory **prepro**, che contiene i file riguardanti il superficiale (ci devono essere anche se non vengono utilizzati, come nel mio caso).

Ora posso far girare il modello restando nella cartella con i file indicati qui sopra e dando da terminale (detto anche “shell”) il comando **./nomefileeseguibile.exe** (ad es., **./cathy.exe**).

Chapter 7

Output file

I file di output sono contenuti nella cartella `simulazioni→output`. Quali file guardare?

- `cumflowvol`: mi mostra il DELTAT a quel time step, il TIME effettivo di simulazione (definito come la somma delle durate dei time step fino a quel NSTEP) e le altre colonne indicano i flussi in uscita/entrata (questo è un output vecchio);
- `dtcoupling`: è legato all'accoppiamento tra superficiale e sotterraneo (posso trascurarlo). Contiene 22 colonne, il cui contenuto è esplicitato all'inizio del file stesso;
- `grid2d.exp`: la prima riga contiene il numero di triangoli nella mesh 2-D (NTRI) ed il numero di nodi nella mesh 2-D (NNOD) (i numeri 1 e 0 sono inseriti dal format e non so cosa indichino). Poi, per ciascun nodo, è indicato il numero identificativo con le coordinate x e y ; in seguito sono riportati il numero identificativo di ciascun triangolo ed i nodi che lo compongono (l'ultima colonna non so cosa indichi, ma il punto è inserito da comando format). Viene scritto solo se IPRT1=3 nel file di input `parm`;
- `grid3d`: la prima riga contiene il numero di nodi nella mesh 2-D (NNOD), il numero di nodi totale in tutta la mesh 3-D (N) ed il numero totale di elementi tetraedrici (NT). Poi sono elencati i nodi che costituiscono ciascuno

dei NT tetraedri (e la rispettiva zona) e quindi le coordinate degli N nodi.

Viene scritto solo se IPRT1=3 nel file di input **parm**;

– **hgatmsf**:

- ACT.FLUX
- OVL.FLUX: è il flusso superficiale, ovvero la differenza tra POT.FLUX e ACT.FLUX;
- RET.FLUX: flusso che dal sotterraneo va al superficiale (return flux);

Da qui io vedo se quello che entra è ciò che mi aspetto o meno

- **hgnansfdirdet**: contiene le informazioni sui nodi con condizioni di Dirichlet (“hg” di solito viene usato per indicare “hydrograph”);
- **mbeconv**: mi mostra il volume d’acqua iniziale nel dominio e poi mostra:

- NSTEP;
- DELTAT;
- TIME;
- NLIN ITER.: numero di iterazioni non lineari effettuate. Da qui posso vedere se il risolutore è adeguato o meno al problema che sto affrontando (ad esempio, se ce ne mette troppe potrei aver bisogno di un DELTAT più piccolo, al contrario posso diminuire la tolleranza se ce ne mette troppo poche);
- AVG. LIN ITER.;
- STORE1 e STORE2: indicano la stessa cosa, ovvero una stima del volume d’acqua nel volume simulato. Sono diversi in quanto vengono calcolati in modo diverso: più sono simili, meglio è! Può essere utile plottare TIME vs STORE2, così vedo l’andamento nel tempo dello storage e capisco se serve aggiungere time step;
- DSTORE: mostra di quanto è variato lo storage rispetto al passo precedente;

- CUM.DSTORE: tiene conto di tutte le variazioni cumulative dello storage;
 - VIN: volume in ingresso. Dovrebbe corrispondere alla pioggia · passo temporale · area;
 - CUM.VIN: quanto ha piovuto in tutto;
 - VOUT: volume in uscita dato sia da Dirichlet che dalle condizioni atmosferiche (evaporazione);
 - CUM.VOUT;
 - VIN+VOUT;
 - M.BAL.ERR.: errore di mass balance calcolato come il volume in ingresso + il volume in uscita – lo storage;
 - REL.MBE;
 - CUM.MBE;
 - CUM.ABS (MBE);
- ◊: più la griglia è raffinata e più sono piccoli i time step, più gli errori dovrebbero essere piccoli (è una questione numerica)
- **nansfdir**: stampa le condizioni di Dirichlet ad ogni time step (se richiesto nel file di input **parm**, scegliendo opportunamente il flag IPRT1);
 - **nansfneu**: stampa le condizioni di Neumann ad ogni time step;
 - **pondhead**: indica quali sono i nodi superficiali “ponded”, che si formano se non tutto si infiltra;
 - **psi**: mostra le pressioni ad ogni nodo della mesh ai tempi di output (io sono nel caso idrostatico, ho la falda a -2.0 m di profondità rispetto al piano campagna e quindi in superficie la pressione su ciascun nodo è pari a -2.00). In questo caso, gli output non avvengono in corrispondenza di ciascun detailed output, ma quando la simulazione supera quel time. Nel data assimilation, invece, gli output coincidono con il time;

- **psisurf**: contiene le pressioni ai nodi superficiali;
- **recharge**
- **risul**: è il file più importante. Contiene un riassunto di quanto è stato mostrato a terminale durante la simulazione. All'inizio contiene un riassunto di alcuni dei vari parametri del modello (contenuti in **parm**, **soil**, ecc.) e poi le informazioni sui vari time step. In particolare, è utile controllare “a campione”:
 - Numero di iterazioni per la LINEAR SOLVER CONVERGENCE (i.e. gradiente coniugato). Se la convergenza è raggiunta, i due residui (residual and real residual) devono essere uguali;
 - Numero di iterazioni per la NON LINEAR CONVERGENCE (i.e. Picard). In particolare:

$$PIKMAX = \|\psi^{t+1} - \psi^t\|_\infty \quad (7.1)$$

$$PL2 = \|\psi^{t+1} - \psi^t\|_2 \quad (7.2)$$

La differenza tra il valore ottenuto da queste due norme (norma infinito e norma euclidea) ed i valori di FINF e FL2 deve essere, al massimo, di 4-5 ordini di grandezza rispettivamente;

- C F = current flux, P F = previous flux, VOL = volume. Questi valori riassumono tutti i flussi in atto come atmospherics, non-atmospherics non-seepage face boundary conditions (e.g. IA DIRIC= Inflow dovuto a Atmospheric Dirichlet). I valori di C F non devono oscillare (cioè variare troppo) da un time step al successivo. Inoltre cioè che a questo time step è C F, al prossimo diventerà P F e così via.

Questo file mi dice se la simulazione non ha girato!!!: a volte mi dice cosa c’è che non va (se l’ha capito), mentre altre volte mi devo arrangiare. Alla fine del file è contenuto un resoconto dell’intera simulazione (quanto c’ha messo, total CPU, ecc.)

- **sw**: di solito contiene la saturazione (tra 0 e 1), ma è stato modificato da Damiano per far stampare in output la water content. Viene stampato solo se IPRT=4. Al momento il codice stampa il water content ($\theta = S_w \cdot \varphi$) ad ogni nodo, ma può essere modificato per stampare la saturazione S ad ogni nodo (basta modificare i commenti alle righe 46 e 47 del file `detout.f` del codice);
- **velelt**: velocità ai tempi di output sugli elementi (“elt”), cioè per ogni elemento stampa la velocità di Darcy;
- **velnod**: è la stessa velocità di **velelt**, ma calcolata per i nodi (vi sono le componenti lungo x , y e z);
- **vp**: contiene l’output dei nodi indicati con NUMVP nel file `parm`.
 - NSTEP: time step index;
 - TIME: time at current time level (time step);
 - SURFACE NODE: numero del nodo (tra quelli indicati in NUMVP) di cui mostra l’output;
 - X: coordinata x del nodo;
 - Y: coordinata y del nodo;
 - Z: coordinata z ;
 - PRESSURE HEAD: profilo idrostatico;
 - SW: water saturation (moisture content/porosity) at each node;
 - CKRW: relative hydraulic conductivity at each node, calcolata con Van Genuchten a quelle pressioni;
- **xyz**: questo file presenta dei contenuti solo se in `parm` è stato messo IPRT1=3 (in caso contrario il file resta bianco). Questo file contiene:
 - NNOD: numero di nodi nella griglia superficiale;
 - N: numero totale di nodi di tutta la mesh 3-D;

- In seguito sono elencati tutti gli N nodi della mesh con le rispettive coordinate x , y e z , a partire dai nodi superficiali, in quanto questi vengono poi replicati sul secondo strato, poi sul terzo e così via. Quindi, se ad esempio io voglio sapere quale nodo è presente 4 strati al di sotto del nodo 3 devo sommare al suo numero 4 volte NNOD. As esempio:

	729	73629	NNOD	N
1	0.00E+00	1.30E+00	0.00E+00	
2	0.00E+00	1.25E+00	0.00E+00	

Se io volessi trovare il nodo che si trova esattamente 4 strati sotto al nodo 2 dovrò fare: $2 + 4 \cdot 729 = 2918$. Infatti il nodo 2918 ha le coordinate (0.00E+00; 1.25E+00; -4.80E+00).

7.1 File di output che si possono trascurare

I seguenti file di output possono essere trascurati (anche se vengono scritti da CATHY):

- `ckrw`
- `debug`
- `enpsia`
- `enpsif`
- `enqoutlet`
- `ensemble`
- `ensubvol`
- `hgflag`
- `hgnansf`

- `hgnudging`
- `hgraph`
- `hgsfdet`
- `iter`
- `net.ris`
- `peatdef`
- `satsurf`
- `sfflag`
- `swsurf`
- `tsnudging`
- `watertable`

Chapter 8

Data assimilation

Il Data Assimilation (DA) consente di inserire in CATHY le misure di elettrica effettuate in campagna e va a correggere le variabili interne a CATHY. Ad esempio, confronto mediante plot la pressione vera (misurata in situ in un pozzetto) e quella simulata da CATHY.

◊: le misure di geoelettrica vengono considerate istantanee anche se in campagna durano una ventina di minuti o più.

Il funzionamento del Data Assimilation è rappresentato in Fig. 8.1. La curva nera rappresenta il vero andamento della pressione in situ, mentre la curva azzurra rappresenta una simulazione del CATHY. L'assimilazione del dato geoelettrico avviene ai tempi indicati dai numeri lungo l'asse t (linee verticali tratteggiate).

Dentro al DA c'è un *Monte Carlo*, cioè vi sono diverse simulazioni di CATHY con differenti parametri del suolo e forzanti atmosferiche: in questo modo si arriva a t_1 con tanti possibili valori di pressione (diverse curve colorate e tratteggiate), i quali dipendono da quello che è successo “in mezzo” più o meno casualmente, in quanto siamo noi a fornire le distribuzioni di probabilità.

Al t_1 c'è l'osservazione che dovrebbe guidare verso il dato reale: essa serve a correggere i valori in modo da avvicinare le diverse realizzazioni all'andamento reale (i.e., la variabilità decresce molto e le realizzazioni dovrebbero addensarsi

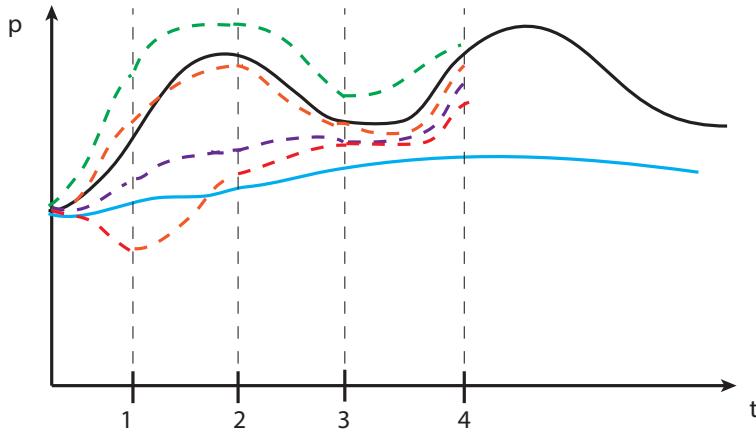


Fig. 8.1: Data assimilation. La curva nera rappresenta il vero andamento della pressione, la curva azzurra quella simulata da CATHY e le curve colorate e tratteggiate rappresentano diverse simulazioni con diversi parametri e forzanti atmosferiche.

attorno al dato reale). Poi, si riparte. Ho a disposizione al t_1 le diverse realizzazioni con i corrispettivi parametri e variabilità e si arriva ad assimilare il secondo dato al tempo t_2 , e così via. Nel caso dell'arancio, per effettuare l'update con il metodo SIR. Come risultato, dovrei iniziare a vedere l'andamento reale ([4], [6]). Si è visto che piuttosto di usare 200 realizzazioni per cercarsi il parametro, a volte è più conveniente usare meno realizzazioni ed effettuare l'assimilazione più volte, partendo dall'ultimo valore. Questo dipende dalla durata della simulazione, da quanti nodi si hanno, ecc.: in generale, più simulazioni Monte Carlo si fanno, meglio è.

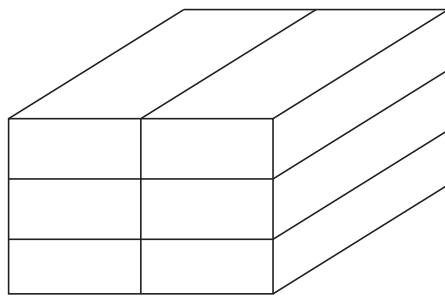


Fig. 8.2: Esempio di zonazione a blocchi.

Un'altra ottica è quella del problema inverso, cioè di usare le misure non solo per modificare le previsioni di CATHY, ma anche per capire come sono fatti i parametri da usare nel modello che non conosciamo (questi parametri, di solito,

non variano nel tempo - ad esempio, θ). Di solito si effettua una zonazione come quella in Fig. 8.2: sta a me capire quanti blocchi servano (più ce ne sono, più è un problema poiché si aggiunge variabilità), così posso stimare i parametri nei diversi blocchi. Per capire se il tutto funziona si sviluppa un caso sintetico, dove cioè conosco già i valori nei vari blocchi. Si effettua una simulazione in cui si prende il dato vero (non è quello misurato!) e vedo se tutta la procedura, prendendo le misure dal dato vero, riporta ai parametri scelti all'inizio. Ad esempio, se ho un valore di conducibilità K vero (linea rossa in Fig. 8.3), parto da un *ensemble* che ha tutti i valori distorti costanti nel tempo fino al t della prima assimilazione, quando i valori variano e si avvicinano al valore vero e così via.

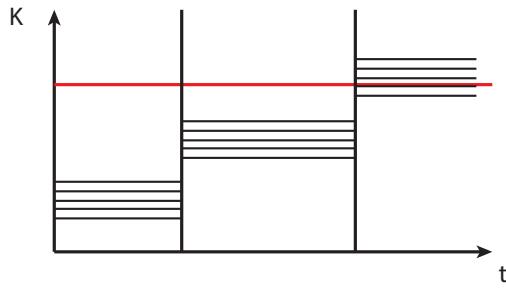


Fig. 8.3: Grafico andamento degli *ensemble* durante l'assimilation.

Inoltre, può essere interessante vedere l'errore sulle misure. Poiché il DA è basato sul metodo Monte Carlo, posso avere un file di output che ad ogni istante indica le misure calcolate dal codice per ogni realizzazione. Erano già stati fatti gli errori associati, riportati nei file “senza nome” (io posso rinominarli). Questi file sono i **fort.xxx**:

- **fort.444**: Real time of output for each Monte Carlo realization. È stato necessario modificare i file di output di CATHY per quando si effettua il Monte Carlo. In CATHY, infatti, c’è un time step del risolutore non lineare che si adatta ai parametri della singola simulazione: non c’è un time step fisso per tutta la simulazione, ma va più veloce o più lento a seconda dei parametri. Questo, però, diventa un problema se si vuole creare un output, poiché non posso richiedere un output ad un certo istante, in quanto quell’istante potrebbe non esistere in quella data simulazione. Per questo

si utilizzano degli istanti in cui sicuramente la simulazione si ferma, cioè quando avviene l'update del DA (in quanto servono tutte le simulazioni per effettuare l'update) e quando vi sono gli output dettagliati (tutte le realizzazioni arrivano a quell'istante e viene effettuato un output di media delle realizzazioni). Altri output potrebbero essere necessari ad ogni passo temporale: nel file `data_ass` è presente il parametro `DTOUT`, il quale indica che ogni volta una realizzazione sorpassa quel step, un output viene generato. In questo file sono riportati i time effettivi di questi output, per ciascuna realizzazione. Ad esempio: l'output è a 2700 s, ma la realizzazione arriva al tempo 2709.84 s;

- `fort.555`: Total ATMPOT for each Monte Carlo realization;
- `fort.556`: Total surface water volume for each Monte Carlo realization;
- `fort.558`: Outflow at the seepage face for each Monte Carlo realization;
- `fort.601`: Ensemble values of the permeability `PERMX`. Questo file è interessante nel caso in cui si debba stimare la conducibilità idraulica. Fornisce i valori di permeabilità dati una zona e riporta una serie di valori statistici ad un certo time. Se avessi n zone, avrei n file. I dati così riportati sono facili da plottare poiché per ciascun tempo ho i valori delle realizzazioni (vedrei il grafico in Fig. 8.3);
- `fort.660`: contiene una possibile misura dell'errore che sto commettendo su un dato nodo. L'errore che si guarda è un errore relativo ed anche l'assimilazione viene fatta sugli errori relativi (per scelta);
- `fort.661`: per tutte le realizzazioni, vengono indicate le misure di geoelettrica per ciascuna realizzazione e per ogni tempo;
- `fort.777`: contiene i vari tempi, e per ciascun tempo contiene le misure e le realizzazioni per ogni misura;
- `fort.778`: contiene, per ogni tempo, gli errori della geoelettrica per le varie realizzazioni. In particolare, qui è riportato l'errore e_1

$$e_1 = \sqrt{\sum_{i=1}^n (y_i^o - y_i)^2} \quad (8.1)$$

- `fort.779`: contiene, per ogni tempo, gli errori e_2

$$e_2 = \frac{e_1}{\sqrt{\sum_{i=1}^n y_i^{o2}}} \quad (8.2)$$

- `fort.780`: contiene, per ogni tempo, gli errori e_3

$$e_3 = \frac{1}{n} \sqrt{\sum_{i=1}^n \frac{(y_i^o - y_i)^2}{y^{o2}}} \quad (8.3)$$

Gli errori e_1 ed e_2 sono delle vere *norme*, mentre e_3 , che è quello usato nel data assimilation, non è una vera norma (è quindi più utile guardare e_3). In altre parole, e_1 sono gli scarti quadratici, e_2 è la media degli errori relativi ed e_3 sono gli scarti quadratici relativi.

8.1 Geoelettrica

Per inserire la geoelettrica è necessario utilizzare nuovi eseguibili aggiuntivi, contenuti nella cartella `ert_code`:

- `ert_grid.exe`: è contenuto nella cartella `ert_code/ert_grid3d` e consente di effettuare tre cose:

1. crea la griglia “allargata” a partire dalla griglia usata da CATHY (da farsi prima di far partire la simulazione);
2. crea dei file di input per la parte del codice che tratta la geoelettrica, in cui mette condizioni di Dirichlet nulle in tutti i bordi (escluso quello superficiale, dove c’è una condizione di no-flow) del nuovo dominio. Si mettono le condizioni di Dirichlet nulle in quanto il potenziale all’infinito lo si assume nullo;

3. questo eseguibile viene utilizzato nella comunicazione tra CATHY ed il programma che calcola i potenziali elettrici. CATHY, infatti, produce la resistività sulla griglia piccola (attraverso Archie) salvando i valori in un file di output. L'eseguibile `ert_grid.exe` prende quindi i dati da questo file di output e crea un file di input con i valori nuovi per ogni elemento della griglia grande con il corrispettivo valore di resistività. Inoltre crea un “link” tra `elec_nodes` (che contiene i nodi su cui si trovano gli elettrodi nella griglia di CATHY) ed i corrispettivi nodi nella nuova griglia;
- `ert_sat3d.exe`: è contenuto nella cartella `ert_code/ert_sat3d` ed ha questo nome in quanto è lo stesso codice che viene usato per le simulazioni di flusso saturo, risolvendo un’equazione ellittica come quella per la geoelettrica. Si assumono 2 nodi nei quali si inietta corrente e quindi si calcola il potenziale elettrico su tutto il dominio (cioè su tutti i nodi) a partire dalla resistività fornita da `ert_grid.exe`. In particolare, uno degli output di questo eseguibile sarà quello che contiene i potenziali in corrispondenza dei nodi dove abbiamo gli elettrodi.

8.2 Eseguibili

Ora mi sposto nella cartella `simulaz_data_ass/ert`, la quale contiene le cartelle `ert`, `input`, `output`, `prepro`, l'eseguibile `cathy.exe` ed il file `cathy.fnames`.

8.2.1 `ert_grid.exe`

Input

Innanzitutto mi sposto nella cartella `ert/input_ert`, dove trovo i seguenti file:

- `datin_grid`:
 - NROW NCOL: numero di righe e di colonne della griglia superficiale;

- AUGMX AUGMY AUGMZ: indico di quanto voglio aumentare la mesh in x , y e z . In particolare, le dimensioni sono aumentate di un numero di nodi pari a $2 \cdot \text{AUGMX}$ lungo l'asse x , $2 \cdot \text{AUGMY}$ lungo l'asse y e AUGMZ lungo z . L'aumento viene effettuato lungo x nei due versi, lungo y nei due versi, lungo z solo verso il basso ed ha luogo con il raddoppio progressivo dell'ampiezza di cella;
- FLAG: può avere due valori:
 - 0: generate the grid for sat3d. **La prima volta che faccio girare il programma uso questo valore per generare la griglia**, poi quando vedo che la griglia va bene faccio girare CATHY;
 - 1: dopo aver fatto girare CATHY metto questo valore nel flag;
- DIRECTION: posso scegliere 0 o 1. Quando faccio girare l'eseguibile, a terminale sarà lui a dirmi se mettere 0 o 1, quindi dovrò poi cambiare e mettere il valore che mi dice lui (e quindi far girare nuovamente?). Devo però fare attenzione che metta sempre lo stesso valore ogni volta che scrive DIRECTION a terminale;
- DIRIC: scelgo come mettere le condizioni di Dirichlet (secondo Damiano sarebbe più corretto mettere 0).

Inoltre, pongo IPRT1=3 nel file di input di CATHY `parm`, in modo da scrivere all'interno del file di output `xyz` facendo girare l'eseguibile `cathy.exe` (NB: i file di input di CATHY sono contenuti nella cartella `simulaz_data_ass/input`).

File in `input_ert` da trascurare

Nella cartella `input_ert` posso trascurare:

- `constitut_tab`
- `ic`
- `material`
- `parm`

– `plotmtv`

– `reservoir`

Esecuzione del programma

Copio l'eseguibile `ert_grid.exe`^{*}, contenuto nella cartella `ert_grid3d`, all'interno della cartella `ert`, la quale contiene le sottocartelle `input_ert`, `output_ert` e `risp`. Nel file `datin_grid` nella cartella `input_ert` setto i vari parametri ma la prima volta che faccio girare questo eseguibile pongo `FLAG = 0`. In questo modo, a terminale, l'eseguibile fornisce le informazioni sulla nuova griglia ampliata.

Output

L'eseguibile `ert_grid.exe` crea innanzitutto un output a terminale elencando i seguenti parametri:

- DIRECTION: indica l'ordine in cui vengono scritti i nodi;
- LENGTH_X, LENGTH_Y, LENGTH_Z LARGE GRID: indica di quanto è aumentata la griglia grande;
- N, NNOD - SMALL GRID: numero di nodi nella mesh 3-D piccola e numero di nodi nella mesh 2-D piccola;
- NSTR - SMALL GRID: numero di strati nella mesh 3-D piccola;
- NCOL, NROW - SMALL GRID: numero di colonne e righe nella mesh piccola;
- BIGN, BIGNT, BIGNNOD: rispettivamente numero di nodi nella mesh 3-D grande, numero di tetraedri nella mesh 3-D grande e numero di nodi nella mesh 2-D grande;

^{*}Per essere generato, richiede anche il file `CATHY.H`, contenente i dimensionamenti massimi: si copia quindi quello utilizzato per generare l'eseguibile `cathy.exe` nella cartella `ert_code/ert_grid3d`

- BIGNSTR, BIGNTRI: numero di strati nella mesh grande, numero di triangoli nella mesh 2-D grande;
- BIGNCOL, BIGNROW: rispettivamente numero di colonne e numero di righe nella mesh grande;

Gli altri output di `ert_grid.exe` su file sono:

- `100.vtk`, nella cartella `simulaz_data_ass/ert`: contiene la nuova griglia;
- `grid_link_nd.dat`, nella cartella `simulaz_data_ass/ert/input_ert`: mostra il passaggio dalla griglia piccola alla grande. Per ciascun nodo è indicato il suo numero nella griglia piccola, in quella grande e le coordinate in entrambe le griglie (x , y e z). Le coordinate restano le stesse, ma è sempre bene controllarle;
- `grid`, nella cartella `simulaz_data_ass/ert/input_ert`: contiene la nuova griglia grande indicando:
 - N^\dagger : numero di nodi;
 - NT^\ddagger : numero di elementi tetraedrici;
 - $NSTR$: numero di strati;
 - $NZONE$: numero di zone;
 - $N1$: maximum number of element connections to a node;
 - Elenco di tutti gli elementi con:
 - Prima colonna: numero identificativo;
 - Dalla seconda alla quinta colonna: nodi che costituiscono quell'elemento;
 - Sesta colonna: strato di appartenenza;
 - Settima colonna: zona di appartenenza
 - Elenco di tutti i nodi con:
 - Prima colonna: numero identificativo del nodo;

[†]Il format $2I7$ è del tipo $rIw.m$, cioè il format $Iw.m$ è ripetuto r volte

[‡]Si veda la nota [†]

- Dalla seconda alla quarta colonna: coordinate x , y e z di quel nodo.
- `elec.dat`: contiene le nuove posizioni degli elettrodi nella griglia grande indicando il numero di elettrodi totali e, per ciascun elettrodo, il numero di nodo ed il corrispettivo numero di elettrodo;
- `dir`: contiene le condizioni di Dirichlet;

Ora che ho creato la griglia e che ho verificato che mi va bene, posso passare a creare le “geoelettriche”. Riprendo quindi il file `datin.grid` e metto `FLAG = 1`, così non modifica più la griglia creata prima ma utilizza i file creati precedentemente per far comunicare i nodi. Poi, considero l’output di CATHY `el_conductivity` (generato dalla subroutine `ert_measure.f` nella cartella `ert`), il quale però è stampato in binario poiché è un file molto grande (se necessario, è possibile stamparlo non in binario): esso contiene i valori di resistività (ho il dubbio che invece sia la conduttività) nodo per nodo. L’eseguibile `ert_grid.exe` legge questo file binario e genera il file `conductivity` (anch’esso binario) all’interno della cartella `input_ert`: quest’ultimo file contiene i valori di conduttività su tutti i nodi della griglia grande.

Quindi, una volta posto `FLAG=1` e controllato il file `el_conductivity`, faccio girare nuovamente l’eseguibile `ert_grid.exe`.

8.2.2 `ert_sat3d.exe`

Input ed esecuzione

Ora posso far girare `ert_sat3d.exe`, il quale produce il campo di potenziale per quella data simulazione. Innanzitutto devo creare l’eseguibile (se non è già presente) spostandomi nella cartella `ert_code/ert_sat3d` e quindi copiarlo nella cartella `simulaz_data_ass/ert`

Output

L’eseguibile `ert_sat3d.exe` genera i seguenti output:

- `501.vtk`: permette di rappresentare le curve di livello dei potenziali elettrici;
- `400.vtk`: probabilmente rappresenta i valori di permeabilità o le condizioni al contorno;
- `fort.1001`: dati due elettrodi di iniezione di corrente, mostra il valore di potenziale calcolato per ciascuna iniezione, in quanto ogni iniezione corrisponde ad una differente coppia di elettrodi di potenziale (mentre gli elettrodi di corrente restano fissi);

8.3 Script

Ora lanciamo uno script che va a cambiare gli elettrodi di iniezione e quindi salva, per ciascuna iniezione, il corrispettivo file `fort.1001`, rinominandolo `nris` (con `n=1,NINJ`) e salvandolo nella cartella `risp`. Lo script da lanciare è il file `prot.sh` (“sh” vuol dire “script”), il quale contiene dei comandi che vengono eseguiti automaticamente dalla shell come se li digitassi io. Cosa contiene questo script?

`cd ert/`. Questo comando deve essere presente quando lo script viene eseguito automaticamente da CATHY, ma deve essere rimosso se si lancia lo script da terminale (se lo script si trova già nella cartella `ert`);

`./ert_grid.exe` esegue il file `ert_grid.exe` che traduce la conducibilità dalla griglia piccola a quella grande, in quanto l’ingrandimento della mesh è già stato effettuato;

`NINJ=“3”` indica che vi sono 3 prove di iniezione di corrente. Questo valore deve essere modificato dall’utente e può essere determinato con il programma `con_inj.exe` (si veda ♦);

`I=“1”` indica che parto dalla prima iniezione;

`echo “1” » input_ert/nrow_ert.dat` salva nel file `nrow_ert.dat` che sono alla prima iniezione;

./pre_ert.exe è un eseguibile che utilizza questi file:

- **datin_ert**
- **config_ert.txt**: da questo legge solo in quali elettrodi si inietta corrente. La “seconda iniezione” ha luogo quando cambiano gli elettrodi di iniezione (ad esempio, si passa da A=1 e B=2 a A=2 e B=3);

Quali file genera?

- **neu**: in questo file scrive “2 NQ” in quanto si inietta sempre in 2 elettrodi, il numero dei nodi in cui avviene l’iniezione, il tempo e l’intensità di corrente;

./ert_sat3d.exe fa girare questo eseguibile così trovo i potenziali;

mv fort.1001 risp/“\${I}”ris sposta il file **fort.1001** (che contiene i potenziali) nella cartella **risp**, chiamando ciascun file **1ris**, **2ris** e così via. Questi nuovi file sono le risposte alle iniezioni;

mv 501.vtk visit“\${I}”.vtk salva anche i **.vtk** così li posso plottare se necessario;

I=\$((I+1)) aggiorna la variabile I (vi devono essere le parentesi tonde, altrimenti questo aggiornamento non avviene e lo script dà errore);

./protocol.exe fa girare questo eseguibile

◆: lo script ripete questo ciclo per NINJ volte. Per contare il numero di iniezioni posso farlo a mano o con un programmino (NB: si definisce "iniezione" l’insieme di volte in cui A e B restano uguali nel file **config_ert** al variare degli elettrodi M e N. Quando variano A, B o entrambi allora ho una nuova iniezione): Damiano usa l’eseguibile **cont_inj.exe**. Questo eseguibile si trova nella cartella **postpro_ert** e deve essere copiato, una volta compilato, nella cartella **input_ert**. I file di input necessari sono **datin_ert** e **config_ert.txt**: il primo deve contenere il numero di misure NMEAS, il numero di elettrodi NELECT e l’intensità di corrente Q (il numero di iniezioni lo si inserisce dopo aver fatto girare l’eseguibile),

il secondo contiene tutti gli NMEAS quadripoli ordinati che hanno superato la soglia d'errore prefissata (i.e., i quadripoli contenuti nel file `protocol.dat` di Binley). Così, l'eseguibile può calcolare il numero di iniezioni, mostrandolo poi a terminale. In alternativa, nella mia versione, il valore NINJ è stampato anche nel file `ninj` (la si trova nella cartella `Eseguibili_Laura`).

◊: tutto questo viene effettuato dallo script, ma per lanciarlo da terminale devo dare `sh ert/prot.sh` o come un comune eseguibile.

◊: la versione che gira sulla WS1 è il `prot.sh` salvato nella cartella `Eseguibili_Laura`.

8.3.1 Eseguibili contenuti nello script

Gli eseguibili devono avere estensione `*.exe` sia nel nome che nello script. I vari codici sorgente si trovano nella cartella `ert_code/postpro_ert` e, una volta che essi sono stati compilati, gli eseguibili devono essere spostati nelle seguenti cartelle:

Esegibile	Cartella
<code>ert_grid.exe</code>	<code>ert</code>
<code>ert_sat3d.exe</code>	<code>ert</code>
<code>pre_ert.exe</code>	<code>input_ert</code>
<code>protocol.exe</code>	<code>risp</code>

In particolare, l'eseguibile `pre_ert.exe` legge i file `datin_ert`, `config_ert.txt`, `nrow_ert.dat` ed `elec.dat` per generare in output i file `neu` e `nrow_new.dat` (quest'ultimo sarà poi cancellato dallo script stesso). L'eseguibile `protocol.exe` legge in input i file `datin_ert`, `config_ert.txt` e `elec.dat`, mentre in output genera i file `protocol.ris` ed `ert.risp`.

8.3.2 Output

Nella cartella `simulaz_data_ass/ert/risp`, questo script non solo colleziona tutti gli output, ma crea anche il file `protocol.ris` grazie all'eseguibile `protocol.exe`.

Così nella cartella `risp` trovo:

- `1ris, 2ris, 3ris, ...`: ne crea uno per ciascuna iniezione. Il file contiene:
 - Q_+ , Q_- : nodi in cui si ha l'iniezione;
 - NMIS: numero della misura;
 - NODE: node in cui si misura il potenziale;
 - ELECT POT: potenziale misurato;
 - RELATIVE ELECT POT: potenziale relativo misurato (poiché iniettiamo sempre 1, questo valore è uguale a ELEC POT);
- `ert.risp`: sarà letto da CATHY e contiene l'elenco delle differenze di potenziale;
- `protocol.ris`: è il protocol che si usa con gli eseguibili di Binley. Infatti contiene:
 - NMEAS: numero della misura;
 - Q_+ Q_- : elettrodi di corrente;
 - P_+ P_- : elettrodi di potenziale;
 - V/I: resistenza;
 - LOG10(V/I): logaritmo in base 10 della resistenza.

8.4 Data Assimilation in CATHY

Le cose viste finora possono essere fatte “a mano”, in modo da verificare ogni singola iniezione dato un certo campo di potenziale generato da CATHY.

In realtà, tutti questi passaggi vengono effettuati da CATHY stesso, grazie alla subroutine `ert_measure` che richiama lo script. CATHY, quindi, ai tempi di output dettagliato, genera le conducibilità elettriche secondo la legge di Archie, definita con i parametri inseriti nel file `archie`, richiama il `protocol.exe` e lo

esegue. Durante la sua esecuzione CATHY si ferma e quando questo eseguibile ha finito CATHY stesso va a leggersi il file **ert.risp** per avere i valori di potenziale simulati (da CATHY).

Per fare questo, però, devo modificare i file di input in CATHY. In particolare:

1. Nel file **parm** devo mettere DAFLAG=1;
2. Devo inserire i parametri della legge di Archie nel file **archie**, contenuto nella cartella **input**. Per farlo vi sono due modi:
 - Inserisco i parametri “a”, “CW”, “m” e “n”;
◊: se $a=0$ allora CATHY usa il secondo metodo
 - Inserisco i rapporti calcolati prendendo un tempo come riferimento, in modo da dover calibrare solo “n”. In realtà, però, per quello di riferimento ho comunque usato i parametri sopra. In particolare, si assumono una conducibilità elettrica $\sigma(t_0)$ ed una saturazione uniformi nel background, così resta da trovare solo “n”;
3. Inserisco il numero di misure ert NERT;
4. Apro il file **data_ass** contenuto nella cartella **input**. Questo file è abbastanza complesso:
 - NENS: lo pongo =1;
 - NOBS: è il numero di misure e deve essere uguale a NERT;
 - NENSMIN: lo metto =1 ma non mi serve;
 - ERT_FLAG: se =0 vuol dire che non c’è ert , se =1 ha luogo il Data Assimilation e quindi devo inserire le mie osservazioni, le quali saranno poi confrontate da CATHY con quelle simulate o se =2 ho un modello diretto, cioè dato un certo problema di flusso voglio vedere cosa ottengo con questi input (per creare il modello diretto devo mettere anche ENKFT=0, vedi più sotto);
◊: per ora metto ERT_FLAG=2. Anche se si hanno le misure è necessario prima confrontare il modello diretto di flusso con le misure

vere. Ad esempio, si può usare l'acquisizione di background per vedere se IC e Van Genuchten vanno bene. Infatti vi sono due problemi:

- (a) Il modello di flusso mi sta dando risultati compatibili con quelli veri?
- (b) I parametri di Archie sono veritieri?
- CORR_FLAG: lo lascio =0
- ENKFT: indico quanti time di assimilation ho (ovvero, quanti time step di acquisizioni ERT ho). Se voglio fare un modello diretto (cioè ERT_FLAG=2), devo mettere ENKFT=0, altrimenti va a leggere comunque i dati inseriti sotto e quindi effettua il DA;
- ENKFTIM: indico il tempo (e.g., in secondi) in cui avviene l'assimilation, cioè quando è stata effettuata la misura ERT;
- NOBS: numero di osservazioni (misure) e corrisponde al numero nel protocol (◊);
- ENKFNOD(I,J): è un elenco che va da 1 a NOBS, mentre il 3 è un flag che indica “electrical potential”. La costruzione di questo parametro è data dal fatto che era stato sviluppato per altri tipi di dati (e.g., pressione), che richiedevano il nodo nella prima colonna ed il flag nella seconda. Poiché le informazioni riguardanti la geoelettrica sono tutte contenute in altri file, qui semplicemente si indica che il dato è di natura elettrica (flag=3), mentre la prima colonna non ha significato;
- DSMEAS: è l'errore (nel caso della geoelettrica è un coefficiente di variazione %). Devo inserire NOBS volte lo stesso valore;
- ENKFVAL: qui inserisco tutte le misure di potenziale (R/I) così come sono indicati e ordinati nel file protocol (◊);

◊: uso il background per capire quali sono le condizioni iniziali (IC) e Van Genuchten e poi non lo assimilo più. Sono simulazioni molto corte poiché non vi è variazione temporale. Voglio semplicemente vedere cosa viene generato da tutta la procedura e quindi confrontarlo con i dati veri.

- ◊: I dati ERT da inserire nel CATHY non sono quelli che escono dal Syscal. Infatti, prima si deve effettuare il controllo dei reciproci e la pulizia del dato sulla base della soglia di errore, poi si devono determinare i common set (in realtà, è possibile effettuare il data assimilation anche senza common set, ma per l'arancio Damiano consiglia di usare i common set)
- ◊: per ogni time di acquisizione ERT devo reinserire NOBS, ENKFNOD(I,J), DSMEAS e ENKFVAL.

8.5 Riassunto dei passaggi da seguire

Per effettuare il Data Assimilation con CATHY è necessario effettuare i seguenti passaggi:

1. Creare un cartella che contenga le sottocartelle `ert`, `input`, `output` e `prepro` (con le relative sottocartelle ed i vari file descritti nel PRIMER), il file `cathy.fnames` e l'eseguibile `cathy.exe`;
2. Nel file `parm` porre IPRT1=3 e far girare da terminale `cathy.exe` in modo da scrivere nell'output `xyz`;
3. Creare la griglia grande ponendo FLAG=0 nel file `datin_grid` ed eseguendo da terminale `ert_grid.exe`;
4. Mettere FLAG=1 nel file `datin_grid` e far girare nuovamente l'eseguibile `ert_grid.exe`, così da creare il file `conductivity` nella cartella `input_ert`;
5. Far girare l'eseguibile `ert_sat3d.exe`;
6. Usare `cont_inj.exe` per vedere quante iniezioni vi sono (NINJ);
7. Modificare lo script `prot.sh` con il valore giusto di NINJ e salvarlo nella cartella `ert`;
8. Modificare nuovamente alcuni parametri nei file di input di CATHY e lanciare l'eseguibile `cathy.exe`

Chapter 9

Modello piante

E disponibile una versione del CATHY consente di creare un modello in cui sia presente l'attività radicale, come descritto in [3].

9.1 Input piante

Per descrivere la presenza e l'attività della pianta si utilizza il file `plant_parm`, il quale contiene i seguenti parametri (e flag) di input:

- NPLANT: numero di piante presenti nel modello;
- NPTYPE: numero di tipi di piante presenti nel modello (se, ad esempio, ho 2 meli e 1 arancio, allora NPLANT=3 e NPTYPE=2);
- ACANOFLAG: lascio 0 (indica se la projected canopy area è data in valore assoluto o riferito all'area nodale);
- PLANT_PRINT: lascio 0 (Indica la tipologia di output);
- NMETEODATA: metto 4 se il LAI (leaf area index) è costante, 5 se invece è funzione del tempo (e quindi viene letto da un altro file di input);
- RHOW: densità (?) dell'acqua (lascio questo valore costante);
- COATM, CCO2STAR e CCO2ATM: sono rispettivamente la concentrazione

di O₂ nell'atmosfera, la concentrazione di riferimento della CO₂ e la concentrazione di CO₂ nell'ambiente;

– SSTOMA:

– ASTOMA:

– TOLLNR:

– ITMAXNR:

– PSILEAF0:

– PSTEP e NPMED:

– Nella riga dove non è indicato alcun parametro, il primo numero indica il nodo su cui insiste quella data pianta ed il secondo indica il tipo di pianta. Quindi avrò tante righe quanto il valore indicato in NPLANT. Ad esempio, ho 3 piante: la prima è nel nodo 6 ed è del tipo 1, la seconda è nel nodo 9 ed è del tipo 2, la terza è nel nodo 15 ed è del tipo 2. In questo caso, avrò NPLANT=3 e NPTYPE=2;

– XMAXP: semi-lunghezza che descrive l'estensione radicale lungo l'asse *x*;

– YMAXP: semi-lunghezza che descrive l'estensione radicale lungo l'asse *y*;

– ZMAXP: lunghezza che descrive l'estensione radicale lungo l'asse *z*;

– VRUX: non utilizzato;

– VRUY: non utilizzato;

– VRUZ: non utilizzato;

– VRUX1: non utilizzato;

– VRUY1: non utilizzato;

– VRUZ1: non utilizzato;

- LAI: valore di leaf area index costante nel tempo;
- GXYLEM_MAX: massima conduttanza dello xilema;
- C_GXYLEM: coefficiente della curva di vulnerabilità;
- D_GXYLEM: coefficiente della curva di vulnerabilità;
- VCMAX: massima capacità di carbossilazione a 25°C;
- KC25: costante di Michaelis per la fissazione della CO₂ a 25°C;
- KO25: costante di Michaelis per la fissazione del O₂ a 25°C;
- COMP25: punto di compensazione della CO₂ a 25°C;
- LA_MAX: maximum marginal water use efficiency;
- LA_BETA: parametro empirico;
- LA_PSILMAX: potenziale idrico fogliare alla massima λ (maximum marginal water efficiency);
- PSILMAX: valore limite utilizzato per capire se la pianta sta lavorando correttamente o meno. Nel caso in cui plottando PSILEAF (vedi 9.2) nel tempo si ottenga come valori minimi PSILMAX, la pianta non sta lavorando correttamente. Molto probabilmente è dovuto alle conduttanze troppo basse;
- LIMIT: apparent quantum yield;
- HCANO: altezza della canopy;
- DATA_LEAF: numero di livelli in cui è suddivisa la canopy;
- ACANO: area proiettata della canopy;
- AXYLEM: area xilematica;
- GROOT_STAR: conduttanza delle radici;

- GSOIL_STAR: conduttanza del suolo;
 - SALT_TOX:
 - NDATA_RDF: numero di righe che il codice leggerà in seguito per determinare la root density function;
 - ZRDF: profondità
 - RDFVAL: valore della root density function alla profondità ZRDF;
- ◆: da XMAXP a SALT_TOX vi saranno tante colonne quanti i tipi di piante, in quanto la prima colonna descrive le piante di tipo 1, la seconda le piante di tipo 2 e così via.

Il secondo file di input consente di inserire i valori che caratterizzano il meteo per il modello che si vuole creare. Questo file, `plant_meteo`, lo si crea con l'eseguibile `out_meteo.exe`, contenuto nella cartella `create_input`. Si devono fornire in input i seguenti file:

- `time.txt`: contiene i time-step a cui sono disponibili gli altri valori dei parametri meteo (ad esempio: 0.0 s, 600.00 s, 1200.0 s, ecc);
- `Temperatura.txt`: contiene i valori di temperatura ai corrispettivi istanti del file `time.txt`;
- `RH.txt`: contiene i valori di umidità relativa ai corrispettivi istanti del file `time.txt`;
- `PAR.txt`: contiene i valori di photosynthetically active radiation ai corrispettivi istanti del file `time.txt`;
- `ZEN.txt`: contiene i valori zenitali ai corrispettivi istanti del file `time.txt`;
- `LAI.txt`: contiene i valori di leaf area index ai corrispettivi istanti del file `time.txt` (letti se NMETEOPLANT=5 nel file `plant_parm`);

Questo eseguibile stampa in output i file `plant_meteo`, `atmbc` e `nansfdirc` da copiare e incollare nella cartella `input`.

Gli ultimi due nuovi file di input, `plant_growth` e `plant_salt`, non li utilizzo.

9.2 Output piante

I file di output presenti sono:

- plant_growth: non lo utilizzo;
- plant_nr: non lo utilizzo;
- plant_leaf: contiene l'output da plottare del modelling della pianta. Contiene undici colonne:
 - TIME = time step del CATHY;
 - TRASP = traspirazione totale calcolata da PSIR. E espressa in m^3/s , quindi per ottenere il valore in m/s deve essere divisa per ACANO;
 - QTOT = traspirazione totale calcolata come somma dei valori nodali QPLANT (come controllo, TRASP dovrebbe essere uguale a QTOT);
 - QTOT_HR = flusso totale di hydraulic redistribution (ovvero i QPLANT negativi, da pianta a suolo);
 - QTOT_TRASP = flusso di root water uptake (ovvero i QPLANT positivi, da suolo a pianta – la traspirazione totale è data dalla somma si hydraulic redistribution e root water uptake, qui si possono guardare le due componenti separate);
 - PSILEAF = leaf water potential (se questa variabile raggiunge il valore minimo indicato da PSILMAX, allora la pianta non sta lavorando correttamente);
 - PSIR = water potential at the base of the plant trunk;
 - PSILMEAN = media di PSILEAF nelle 24 ore precedenti;
 - GSTOMA(15) = stomatal conductance (poiché la “big leaf” è divisa in strati – 30 nel mio caso – qui stampa il valore allo strato 15);
 - FC = flusso di carbonio;
 - LASTOMA = è λ in [3] e [2];

Chapter 10

Rappresentazione degli output: VisIt*

VisIt[†] è un programma open source che consente di rappresentare dati forniti in diversi formati (tra i quali .vtk) sia in 2–D che 3–D.

10.1 Plot di output CATHY

CATHY fornisce diversi output in formato .vtk. Per aprire VisIt è sufficiente digitare da terminale **visit** in qualunque posizione ci si trovi. Quindi cliccare su *File* → *Open File* o cliccare su “Open”, come mostrato nella schermata in Fig. 10.1, per aprire il file desiderato. Attenzione: se “File grouping” è impostato su “On” (freccia blu in Fig. 10.1), allora verranno selezionati ed aperti TUTTI i file con estensione .vtk presenti nella cartella selezionata.

Se, invece, si vuole aprire un determinato file è necessario spostarsi nella cartella contenente quel file e digitare da terminale il comando **visit -o nome-file.vtk**.

◇: Se VisIt non è già stato aperto precedentemente, potrebbe essere selezionata l’opzione “File grouping” nel menù “Open File” (Fig. 10.1). In questo caso, anche se viene specificato il nome del file da aprire, VisIt aprirà tutti i file .vtk

*Guida completa al link: <https://wci.llnl.gov/codes/visit/1.5/VisItUsersManual1.5.pdf>

†Download link: <https://wci.llnl.gov/simulation/computer-codes/visit/>

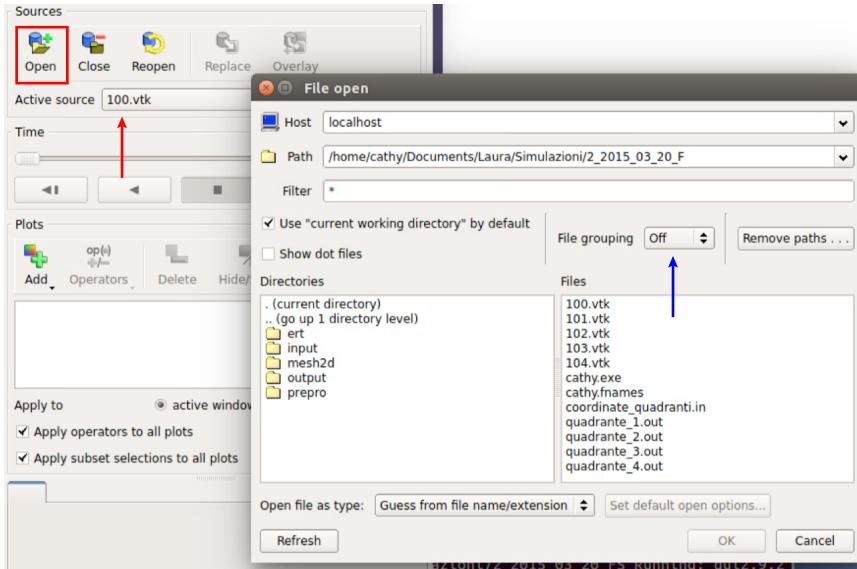


Fig. 10.1: Apertura file con VisIt.

presenti in quella cartella. Se, invece, VisIt è già stato aperto ed è stata deselezionata l'opzione “File grouping”, allora verrà aperto il file indicato nel comando da terminale.

A prescindere dal metodo utilizzato per aprire il/i file desiderato/i con VisIt, è sempre necessario controllare che il nome di tale/i file sia poi indicato come “Active source” (Fig. 10.1). Cliccare quindi su *Add → Pseudocolor → pressure (or saturation)* e quindi su *Draw* (Fig. 10.2). Seguendo lo stesso procedimento è inoltre possibile aggiungere anche la mesh creata con CATHY (*Add → Mesh → mesh* e quindi *Draw*).

Per modificare la scala di colori nel caso di una rappresentazione in pseudocolor, è sufficiente cliccare su *Plot Attributes → Pseudocolor...* nella barra dei menù. Si aprirà una finestra in cui è possibile scegliere:

- Tipo di scala (lineare, logaritmica o con un certo skew);
- Limiti (possono coincidere con quelli del dato originale o essere scelti arbitrariamente);
- Centering;
- Color table (cliccando sulla scala di colori è possibile scegliere tra quelle preesistenti ed è inoltre possibile invertire l'ordine dei colori);

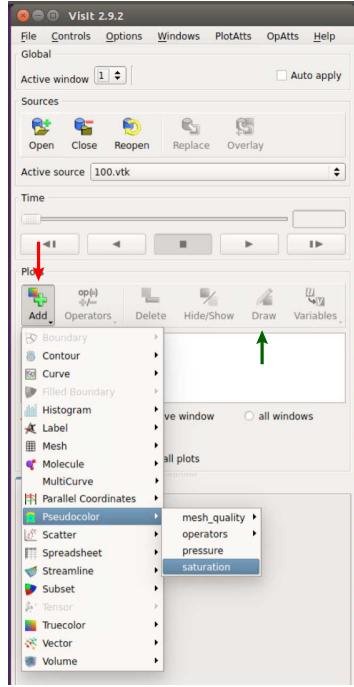


Fig. 10.2: Add dati con VisIt.

- Opacità;
- Geometry (consente di modificare alcune caratteristiche della scala di colori).

Una volta selezionate le opzioni che si preferiscono, cliccare su *Apply*.

Se si vogliono modificare gli assi del proprio grafico, cliccare nella barra dei menu su *Controls* → *Annotation...* e quindi selezionare la tab *2-D* o *3-D* a seconda del tipo di grafico su cui si sta lavorando. Cliccando, invece, sulla tab *Objects* è possibile modificare la posizione e lo stile della legenda.

10.2 ThreeSlice operator

Questo operatore consente di creare tre slice sullo stesso volume, ciascuna perpendicolare ad uno dei tre assi (*x*, *y* e *z*).

Dopo aver inserito il file *.vtk di interesse mediante il comando *Add* → *Pseudocolor*, NON cliccare su *Draw*, ma lasciare l'elemento selezionato (che ora sarà scritto in verde acido su sfondo blu). Cliccare ora su *Operators* → *Slicing* →

ThreeSlice (accanto a *Add*, Fig. 10.2): si noterà che ora l'elemento è sempre riportato in verde acido su sfondo blu, ma la scritta è cambiata: “Pseudocolor – ThreeSlice(xxx)”. Ora cliccare su *Draw* per disegnare i dati nella Window 1. Per modificare la posizione delle slice nel grafico, cliccare sulla freccia accanto a “Pseudocolor – ThreeSlice(xxx)” e quindi cliccare due volte su “ThreeSlice”, in modo da aprire la finestra *ThreeSlice operator attributes*: qui si possono indicare le coordinate x, y e z del punto in cui i tre piani si incrociano.

Selezionando l'opzione ‘Interactive’, è possibile modificare direttamente nella Window 1 la posizione del punto di intersezione delle tre slice, grazie allo strumento “Point tool” (settimo strumento da sinistra nella seconda riga di strumenti della Window 1, Fig. 10.3).

10.3 Isosuperfici

Per creare delle isosuperfici è necessario cliccare su *Operators* → *Slicing* → *Iso-surface* e quindi cliccare su *Draw*. Come mostrato in Fig. 10.3, cliccando sulla freccia verso il basso è possibile vedere quali informazioni sono state plottate nella Window 1. In particolare, con un doppio clic su “Isosurface” si apre la finestra “Isosurface operator attributes”, nella quale è possibile settare i parametri delle isosuperfici, quali i valori massimi e minimi, il numero di livelli presenti, ecc. Una volta definiti, cliccare su *Apply*.

Per modificare l'opacità delle isosuperfici cliccare due volte su pseudocolor (Fig. 10.3), quindi modificare il parametro Opacity (ad esempio, selezionando “Costant” ed un valore pari al 50%). Quindi, cliccare “Apply”.

Inoltre, è possibile aggiungere una (o più) isosuperfici aggiuntive: si aggiunge un nuovo “layer” *Add* → *Pseudocolor* → *pression* (*o saturation*) e quindi si clicca su *Draw*, come visto precedentemente. Per avere più isosuperfici sulla stessa figura è necessario deselezionare l'opzione “Apply operators to all plots” prima di creare le isosuperfici sul secondo layer.

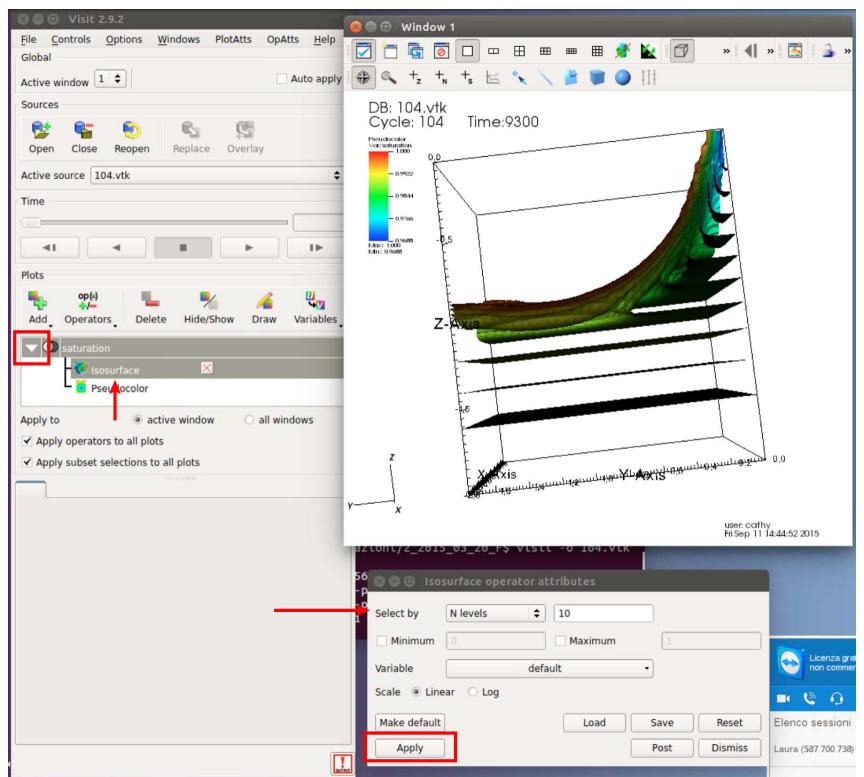


Fig. 10.3: Isosuperfici con VisIt.

Bibliography

- [1] M Camporese, C Paniconi, M Putti, and S Orlandini. Surface-subsurface flow modeling with path-based runoff routing, boundary condition-based coupling, and assimilation of multisource observation data. *Water Resources Research*, 46(2), 2010.
- [2] Gabriel Katul, Stefano Manzoni, Sari Palmroth, and Ram Oren. A stomatal optimization theory to describe the effects of atmospheric CO₂ on leaf photosynthesis and transpiration. *Annals of botany*, 105(3):431–442, 2010.
- [3] Gabriele Manoli, Sara Bonetti, Jean Christophe Domec, Mario Putti, Gabriel Katul, and Marco Marani. Tree root systems competing for soil moisture in a 3D soil-plant model. *Advances in Water Resources*, 66:32–42, 2014.
- [4] Gabriele Manoli, Matteo Rossi, Damiano Pasetto, Rita Deiana, Stefano Ferraris, Giorgio Cassiani, and Mario Putti. An iterative particle filter approach for coupled hydrogeophysical inversion of a controlled infiltration experiment. *Journal of Computational Physics*, 283:37–51, 2015.
- [5] Claudio Paniconi, Marino Marrocù, Mario Putti, and Mark Verbunt. Newtonian nudging for a Richards equation-based distributed hydrological model. *Advances in Water Resources*, 26(2):161–178, 2003.
- [6] Matteo Rossi, Gabriele Manoli, Damiano Pasetto, Rita Deiana, Stefano Ferraris, Mario Putti, and Giorgio Cassiani. Coupled inverse modeling of a controlled irrigation experiment using multiple hydrogeophysical data. *Submitted for publication*, 2014.
- [7] M Th Van Genuchten. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil science society of America journal*, 44(5):892–898, 1980.