

Modellazione idrologica con QGIS model builder e creazione di plugins

- Esercitazione 6 -

Annalisa Minelli, PhD

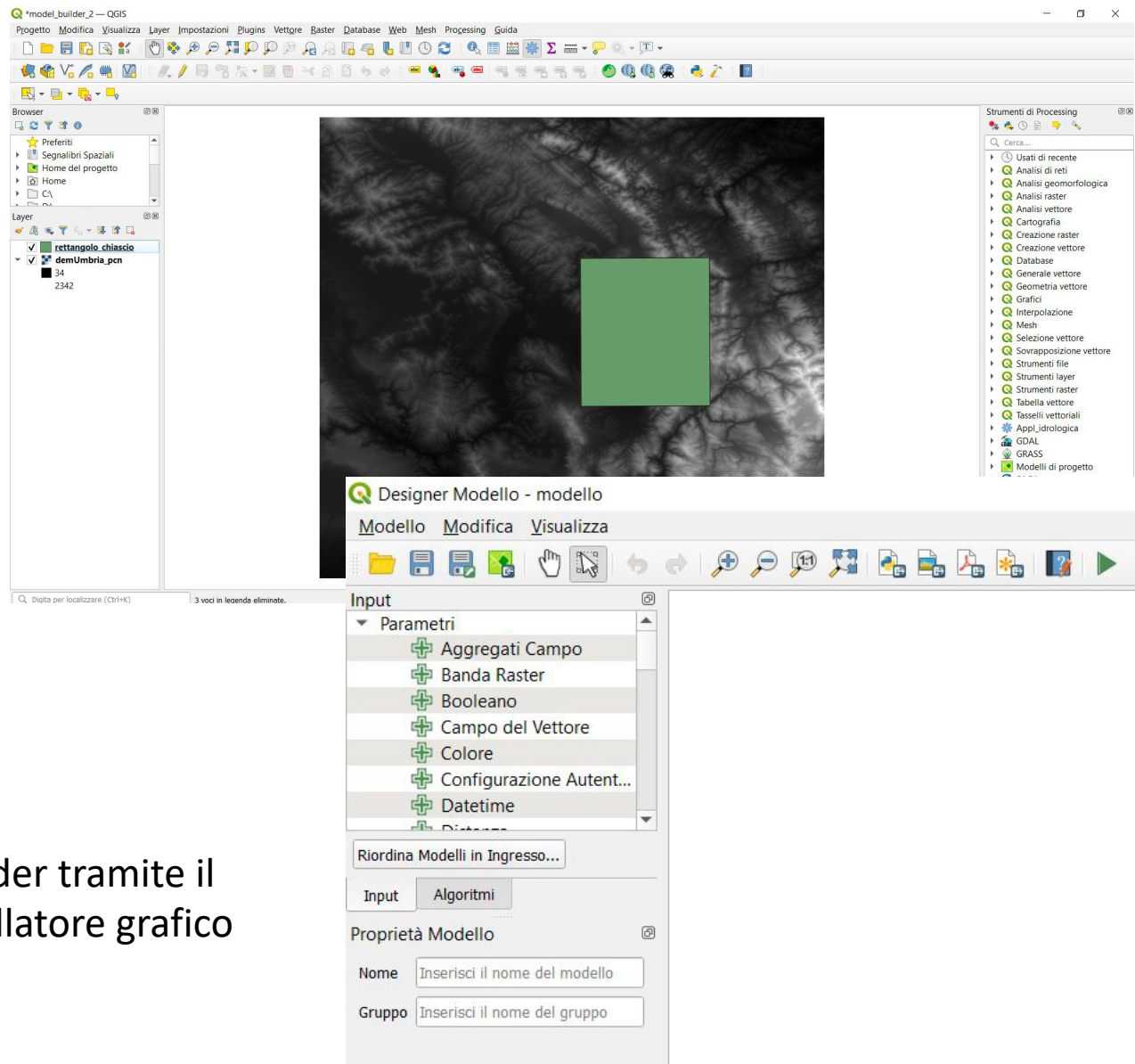
ISPRA

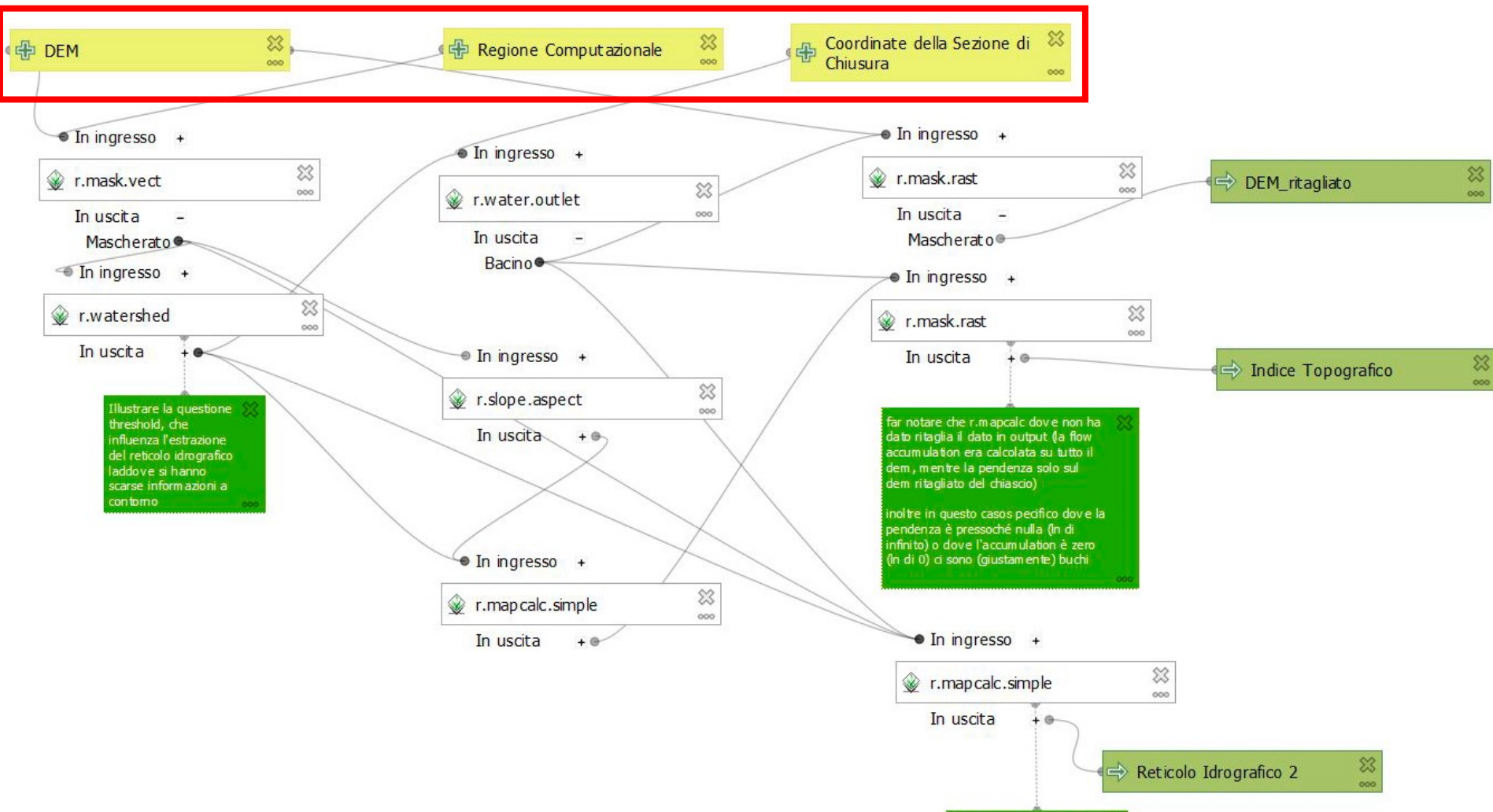
Set up iniziale

Apriamo un nuovo progetto di QGIS e carichiamo i layers:

- demUmbria_pcn.tiff - raster delle quote dell'intera regione Umbria
- rettangolo_chiasco.shp – rettangolo vettoriale che delimita a spanne l'area del bacino del F. Topino

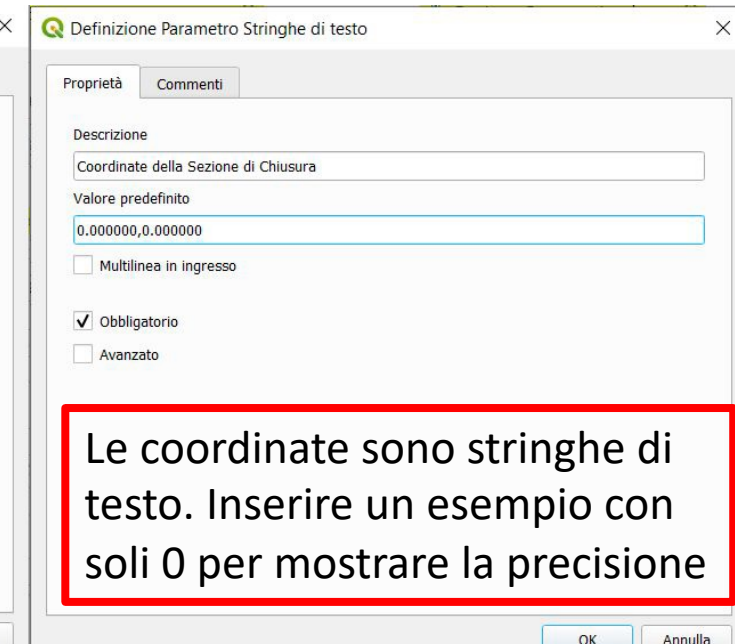
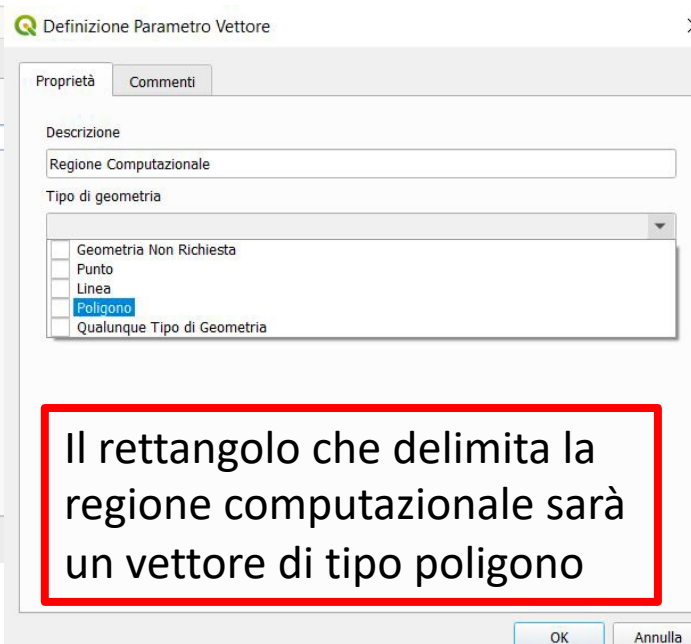
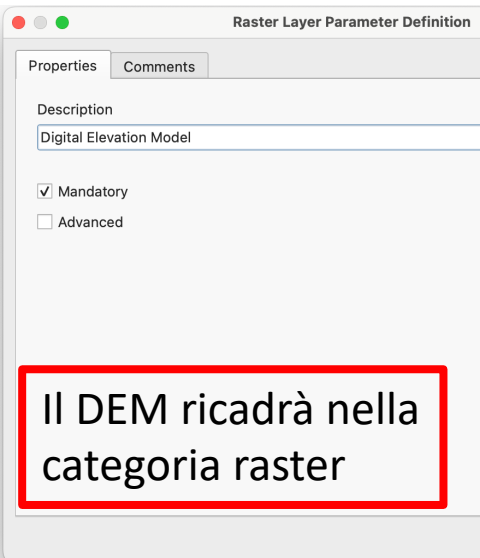
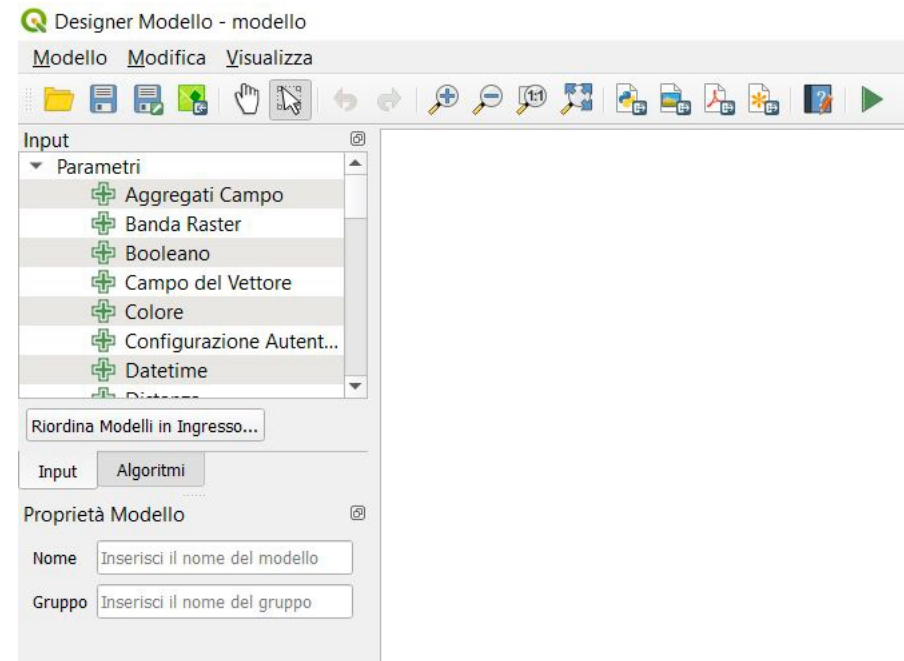
Apriamo quindi il Model Builder tramite il percorso: Processing > Modellatore grafico

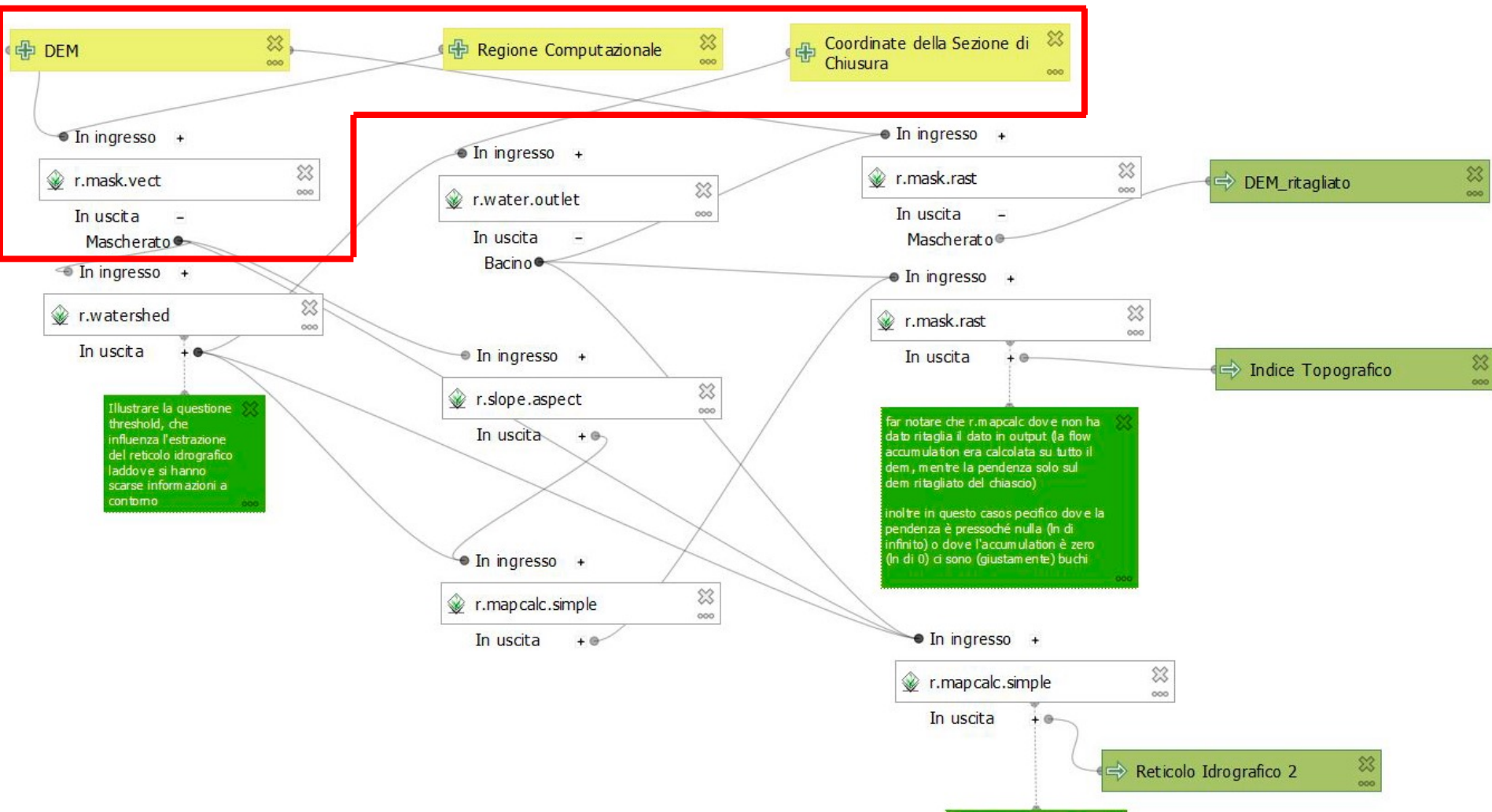




Definire gli input

Gli input al nostro modello saranno, oltre al DEM e al rettangolo che delimita l'area di lavoro, anche una coppia di coordinate che rappresenta la sezione di chiusura del bacino oggetto di studio. Quindi nel pannello a sinistra selezioniamo "Input" e andiamo a prendere gli elementi corrispondenti:

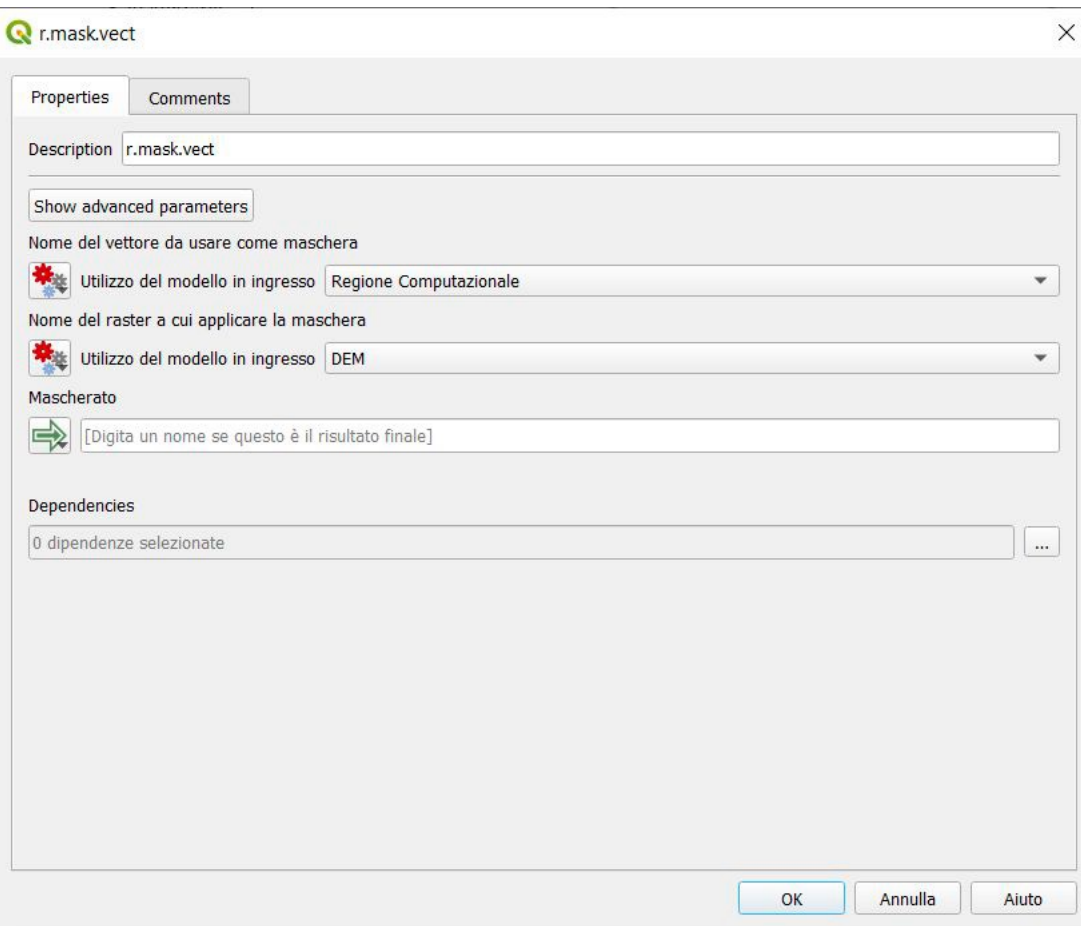




Definire la regione computazionale

Mascheriamo ora il DEM con il vettoriale della regione computazionale. Il modulo di GRASS preposto è r.mask.vect. Lo richiamiamo dalla finestra “Algoritmi”.

Sulle rotelline a fianco dei due parametri richiesti selezioniamo “Modello in ingresso” e quindi selezioniamo rispettivamente:

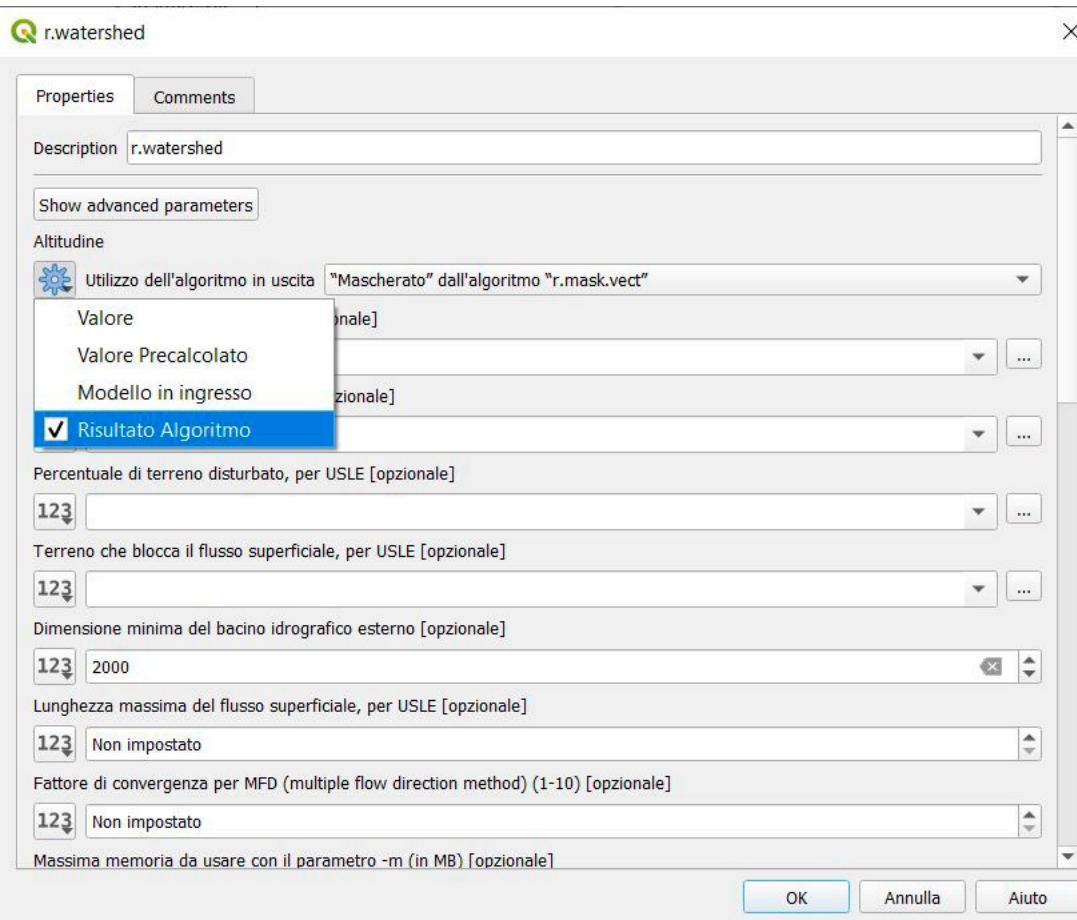


- la regione computazionale come maschera
- il DEM come mappa su cui la maschera avrà effetto

Lasciamo invece vuoto l’output, dal momento che si definisce un nome per l’output solo se la mappa prodotta dal modulo sarà poi tra gli output dell’intero modello (in questo caso no).

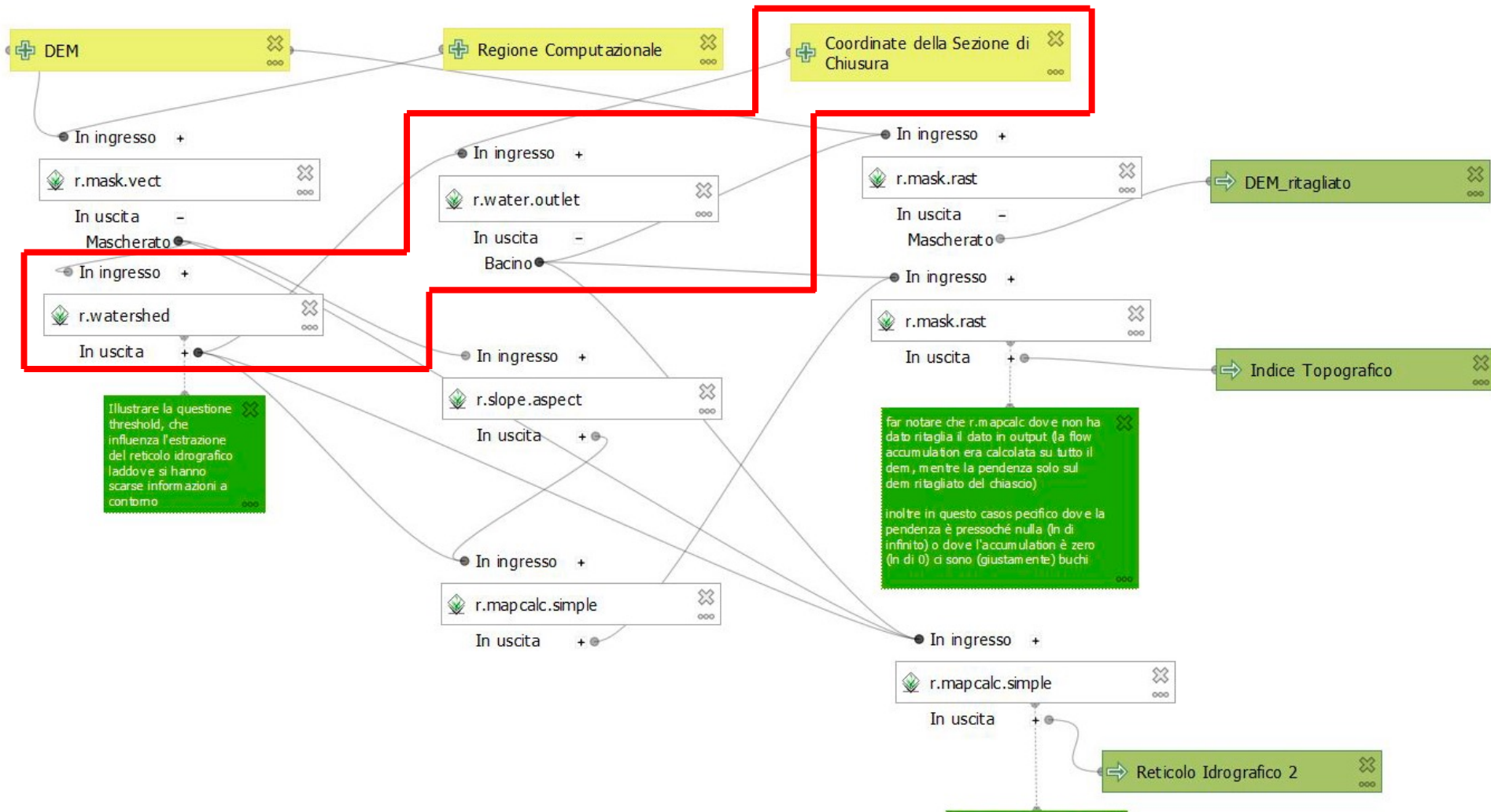
Definire la regione computazionale

Definiamo ora l'area di drenaggio (in numero di celle che drenano acqua all'interno di altre celle), i segmenti del reticolo fluviale e le direzioni di drenaggio tramite r.watershed. Cerchiamo sempre il modulo tramite la scheda "Algoritmi".



In questo caso l'input è il solo DEM mascherato proveniente da r.mask.vect quindi dalle rotelline a fianco al nome del parametro scegliamo "risultato algoritmo".

Inseriamo come soglia un valore di 2000 celle, lasciamo invece bianchi gli output, in quanto non sono output "finali".



Definire il bacino a partire dalle coordinate della sezione di chiusura e dalle direzioni di drenaggio

L'area del bacino sottesa ad una data sezione di chiusura viene definita tramite l'algoritmo **r.water.outlet** a partire dalle coordinate e dalla mappa delle direzioni di drenaggio appena creata con r.watershed. In questo caso il raster delle direzioni di drenaggio è un input di tipo "risultato algoritmo" (di r.watershed) mentre le coordinate sono fornite come input al

modello generale (quindi si seleziona "Modello in ingresso" come tipologia).

E' di fondamentale importanza quando si scelgono le coordinate della sezione di chiusura che esse ricadano nel reticolo estratto da r.watershed (o comunque una zona di vaso naturale a partire dal DEM).

r.water.outlet

Properties Comments

Description r.water.outlet

Show advanced parameters

Nome del raster in ingresso

Utilizzo dell'algoritmo in uscita "Direzione di drenaggio" dall'algoritmo "r.watershed"

Coordinate del punto di presa

Utilizzo del modello in ingresso Coordinate della Sezione di Chiusura

Bacino

[Digita un nome se questo è il risultato finale]

Dependencies

0 dipendenze selezionate

OK Annulla Aiuto

Definire la mappa raster di pendenza del bacino

La mappa delle pendenze viene calcolata in GRASS con l'algoritmo **r.slope.aspect** a partire dal DEM ritagliato.

Richiamando il comando sempre tramite la scheda "Algoritmi" a destra della finestra principale, si può selezionare il DEM ritagliato come "risultato algoritmo" (di r.mask.vect)

e lasciare il resto dei parametri ai valori di default.

Anche in questo caso l'output (mappa di pendenza) non sarà l'output del modello generale quindi non scriveremo nulla in corrispondenza del parametro "Pendenza".

r.slope.aspect

Properties Comments

Description r.slope.aspect

Show advanced parameters

Altitudine

Utilizzo dell'algoritmo in uscita "Mascherato" dall'algoritmo "r.mask.vect"

Formato per il riporto della pendenza [opzionale]

123 degrees

Tipo di esposizione e pendenza del raster in uscita [opzionale]

123 FCELL

Non allineare la regione attuale al layer di elevazione

123 Si

Calcola il risultato ai bordi e vicino ai valori NULL

123 No

Crea l'esposizione in gradi in senso orario dal Nord (azimut), con flat = -9999

123 No

Fattore moltiplicativo per convertire unità elevazione in metri [opzionale]

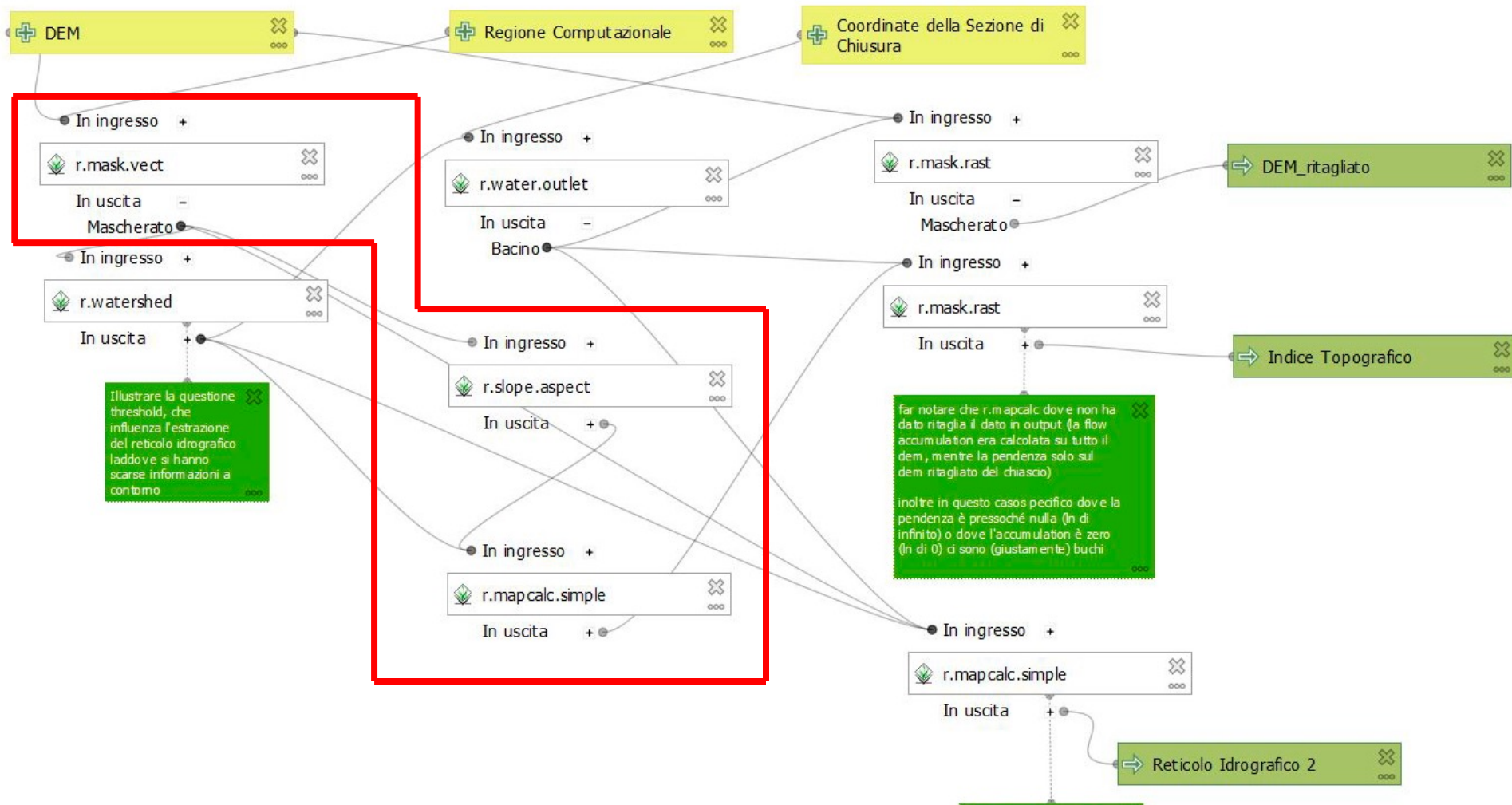
123 1,000000

Minimo valore di pendenza (in percentuale) per cui l'esposizione viene calcolata [opzionale]

123 0,000000

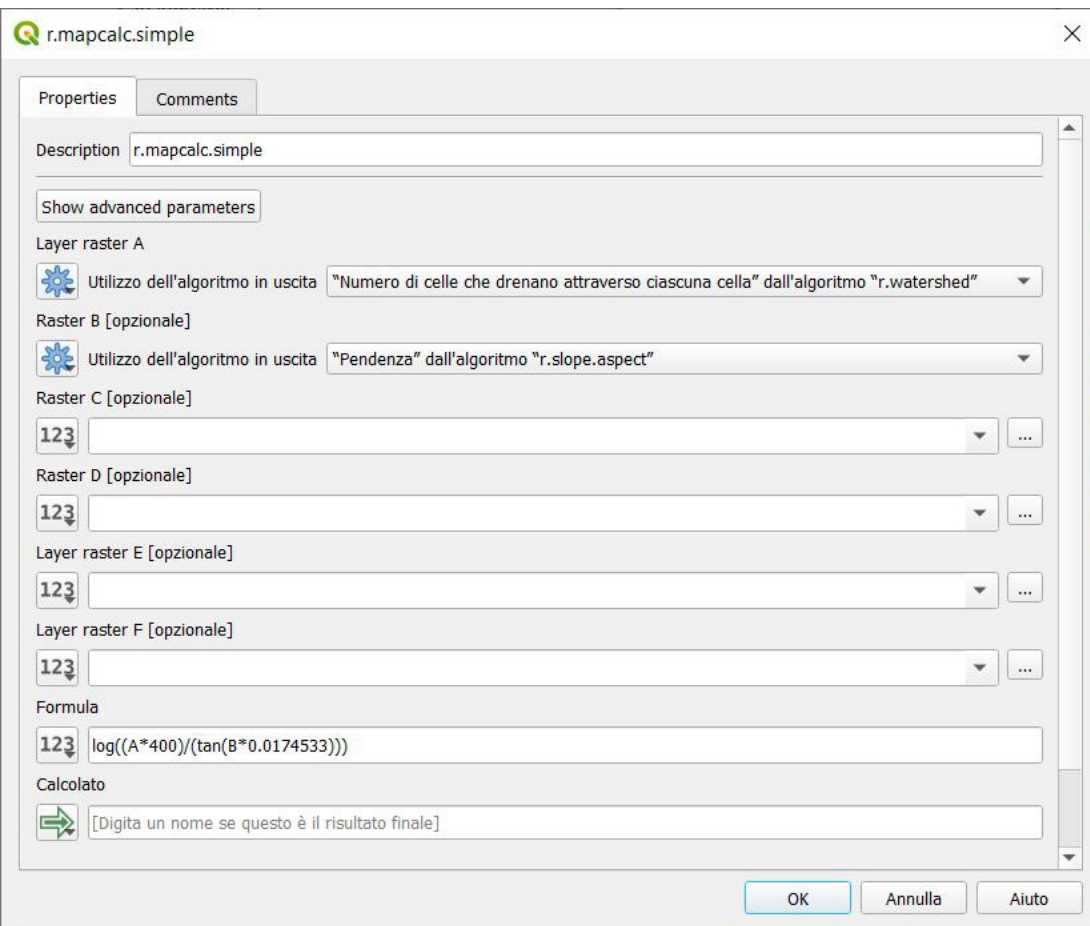
Pendenza [opzionale]

OK Annulla Aiuto



Definire la mappa raster di pendenza del bacino

La mappa dell'indice topografico si ottiene implementando una espressione in r.mapcalc a partire dalla mappa della pendenza (calcolata da r.slope.aspect) e quella della flow accumulation (calcolata da r.watershed).



r.mapcalc.simple

Properties Comments

Description r.mapcalc.simple

Show advanced parameters

Layer raster A
Utilizzo dell' algoritmo in uscita "Numero di celle che drenano attraverso ciascuna cella" dall' algoritmo "r.watershed"

Raster B [opzionale]
Utilizzo dell' algoritmo in uscita "Pendenza" dall' algoritmo "r.slope.aspect"

Raster C [opzionale]
123

Raster D [opzionale]
123

Layer raster E [opzionale]
123

Layer raster F [opzionale]
123

Formula
123 $\log((A*400)/(\tan(B*0.0174533)))$

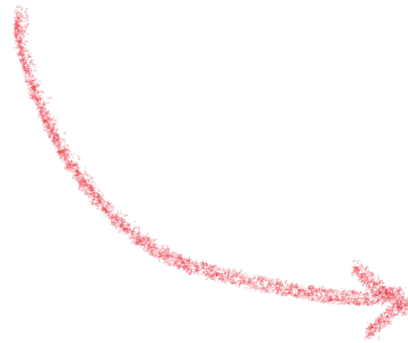
Calcolato
[Digita un nome se questo è il risultato finale]

OK Annulla Aiuto

Richiamiamo quindi sia l'accumulazione (A) che la pendenza (B) come "risultato algoritmo" e cerchiamo, nella sintassi di r.mapcalc (in questo caso è utile andare sull'help), gli operatori che ci servono.

Definizione della mappa dell'indice topografico

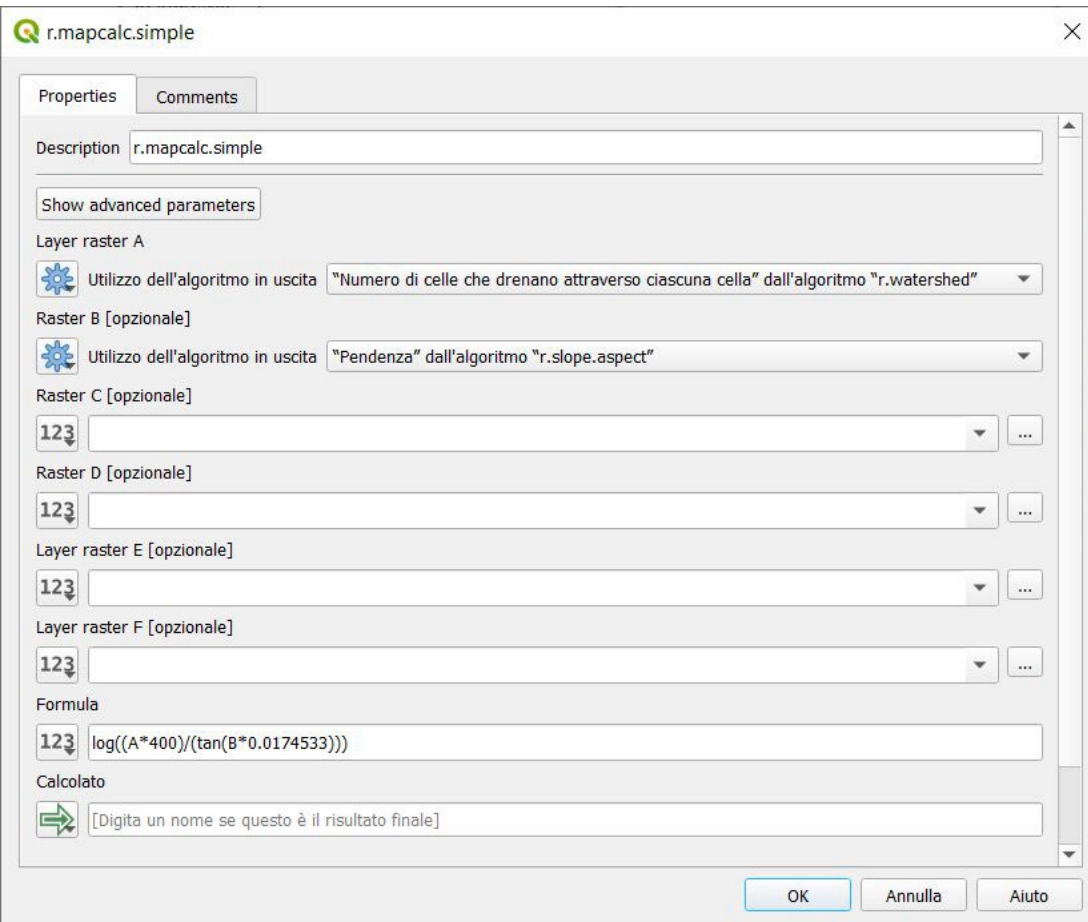
Richiamiamo quindi sia l'accumulazione (A) che la pendenza (B) come "risultato algoritmo" e cerchiamo, nella sintassi di r.mapcalc (in questo caso è utile andare sull'help), gli operatori che ci servono.



function	description	type
abs(x)	return absolute value of x	*
acos(x)	inverse cosine of x (result is in degrees)	F
asin(x)	inverse sine of x (result is in degrees)	F
atan(x)	inverse tangent of x (result is in degrees)	F
atan(x,y)	inverse tangent of y/x (result is in degrees)	F
ceil(x)	the smallest integral value not less than x	*
cos(x)	cosine of x (x is in degrees)	F
double(x)	convert x to double-precision floating point	F
eval([x,y,...],z)	evaluate values of listed expr, pass results to z	
exp(x)	exponential function of x	F
exp(x,y)	x to the power y	F
float(x)	convert x to single-precision floating point	F
floor(x)	the largest integral value not greater than x	*
graph(x,x1,y1[x2,y2..])	convert the x to a y based on points in a graph	F
graph2(x,x1[,x2,..],y1[,y2..])	alternative form of graph()	F
if	decision options:	*
if(x)	1 if x not zero, 0 otherwise	
if(x,a)	a if x not zero, 0 otherwise	
if(x,a,b)	a if x not zero, b otherwise	
if(x,a,b,c)	a if x > 0, b if x is zero, c if x < 0	
int(x)	convert x to integer [truncates]	I
isnull(x)	check if x = NULL	
log(x)	natural log of x	F
log(x,b)	log of x base b	F
max(x,y[,z...])	largest value of those listed	*
median(x,y[,z...])	median value of those listed	*
min(x,y[,z...])	smallest value of those listed	*
mod(x,y)	return the modulus (the remainder) of x/y	*
mode(x,y[,z...])	mode value of those listed	*
nmax(x,y[,z...])	largest value of those listed, excluding NULLs	*
nmedian(x,y[,z...])	median value of those listed, excluding NULLs	*
nmin(x,y[,z...])	smallest value of those listed, excluding NULLs	*
nmode(x,y[,z...])	mode value of those listed, excluding NULLs	*
not(x)	1 if x is zero, 0 otherwise	
pow(x,y)	x to the power y	*
rand(a,b)	random value x : a <= x < b	*
round(x)	round x to nearest integer	I
round(x,y)	round x to nearest multiple of y	
round(x,y,z)	round x to nearest y*i+z for some integer i	
sin(x)	sine of x (x is in degrees)	F
sqrt(x)	square root of x	F
tan(x)	tangent of x (x is in degrees)	F
xor(x,y)	exclusive-or (XOR) of x and y	I

Definizione della mappa dell'indice topografico

La mappa dell'indice topografico si ottiene implementando una espressione in r.mapcalc a partire dalla mappa della pendenza (calcolata da r.slope.aspect) e quella della flow accumulation (calcolata da r.watershed)



The screenshot shows the 'r.mapcalc.simple' dialog box with the following configuration:

- Description: r.mapcalc.simple
- Show advanced parameters: [button]
- Layer raster A: Utilizzo dell'algoritmo in uscita "Numero di celle che drenano attraverso ciascuna cella" dall'algoritmo "r.watershed"
- Raster B [opzionale]: Utilizzo dell'algoritmo in uscita "Pendenza" dall'algoritmo "r.slope.aspect"
- Raster C [opzionale]: 123
- Raster D [opzionale]: 123
- Layer raster E [opzionale]: 123
- Layer raster F [opzionale]: 123
- Formula: $\log((A*400)/(tan(B*0.0174533)))$
- Calcolato: [button]
- [Digita un nome se questo è il risultato finale]

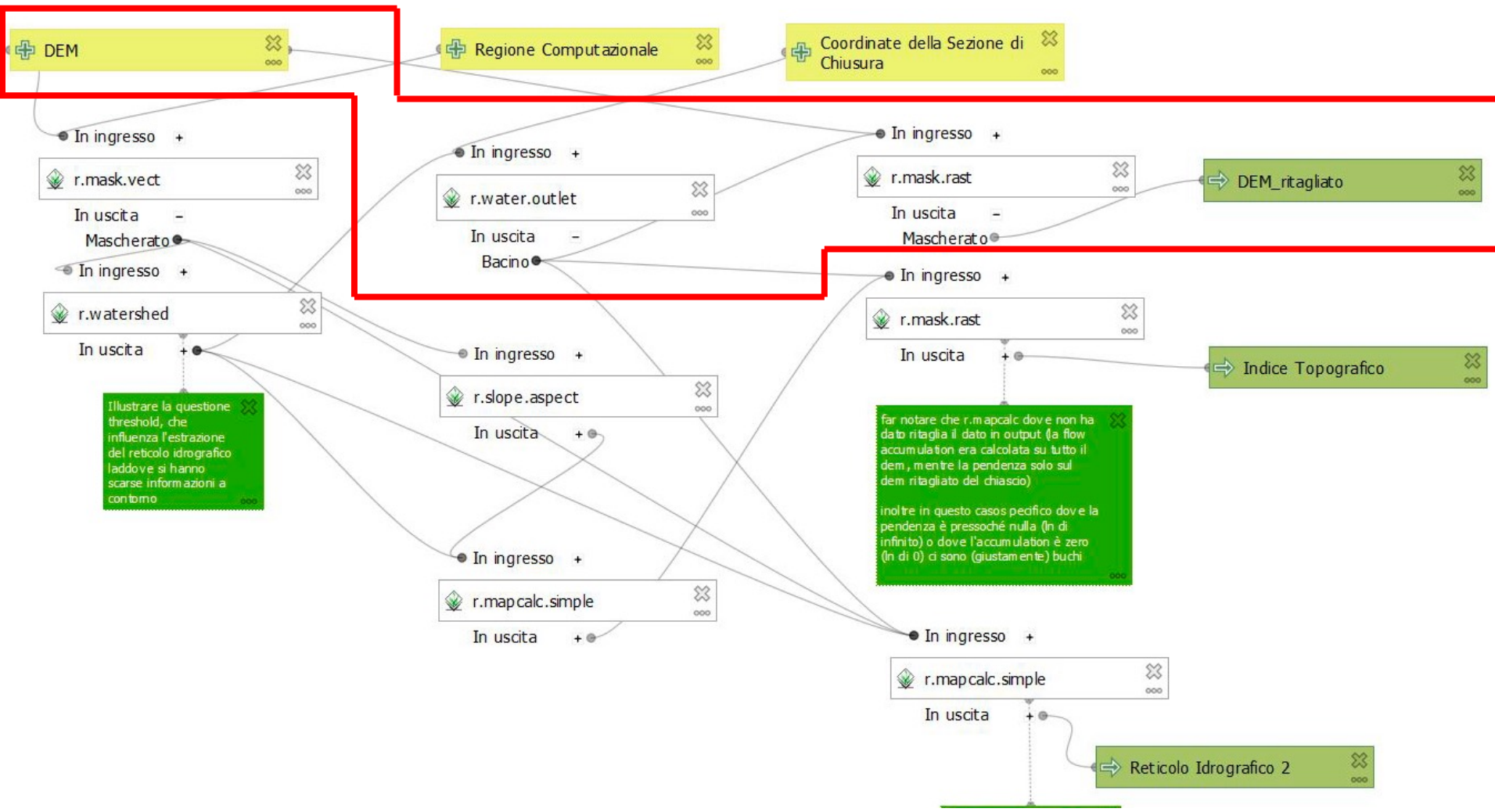
Buttons at the bottom: OK, Annulla, Aiuto

Nel nostro caso la sintassi è:

$$\log((A*400)/\tan(B*0.0174533))$$

Dove:

- log è il log naturale,
- 400 è l'area della cella del raster (che ha risoluzione 20m),
- 0.0174533 è il fattore di conversione da gradi a radianti



Produzione dei layers in output ritagliati: il DEM

Questa e la prossime due operazioni sono volte a “ritagliare” i prodotti finali sull’area del bacino estratta da r.water.outlet. Il DEM lo ritagliamo a partire da quello di partenza, usando r.mask.rast

r.mask.rast

Properties Comments

Description r.mask.rast

Show advanced parameters

Nome del raster da usare come maschera

Utilizzo dell'algoritmo in uscita "Bacino" dall'algoritmo "r.water.outlet"

Nome del raster a cui applicare la maschera

Utilizzo del modello in ingresso DEM

Valori raster da usare per maschera. Formato: 1 2 3 a 7 * [opzionale]

123 *

Mascherato

DEM_ritagliato

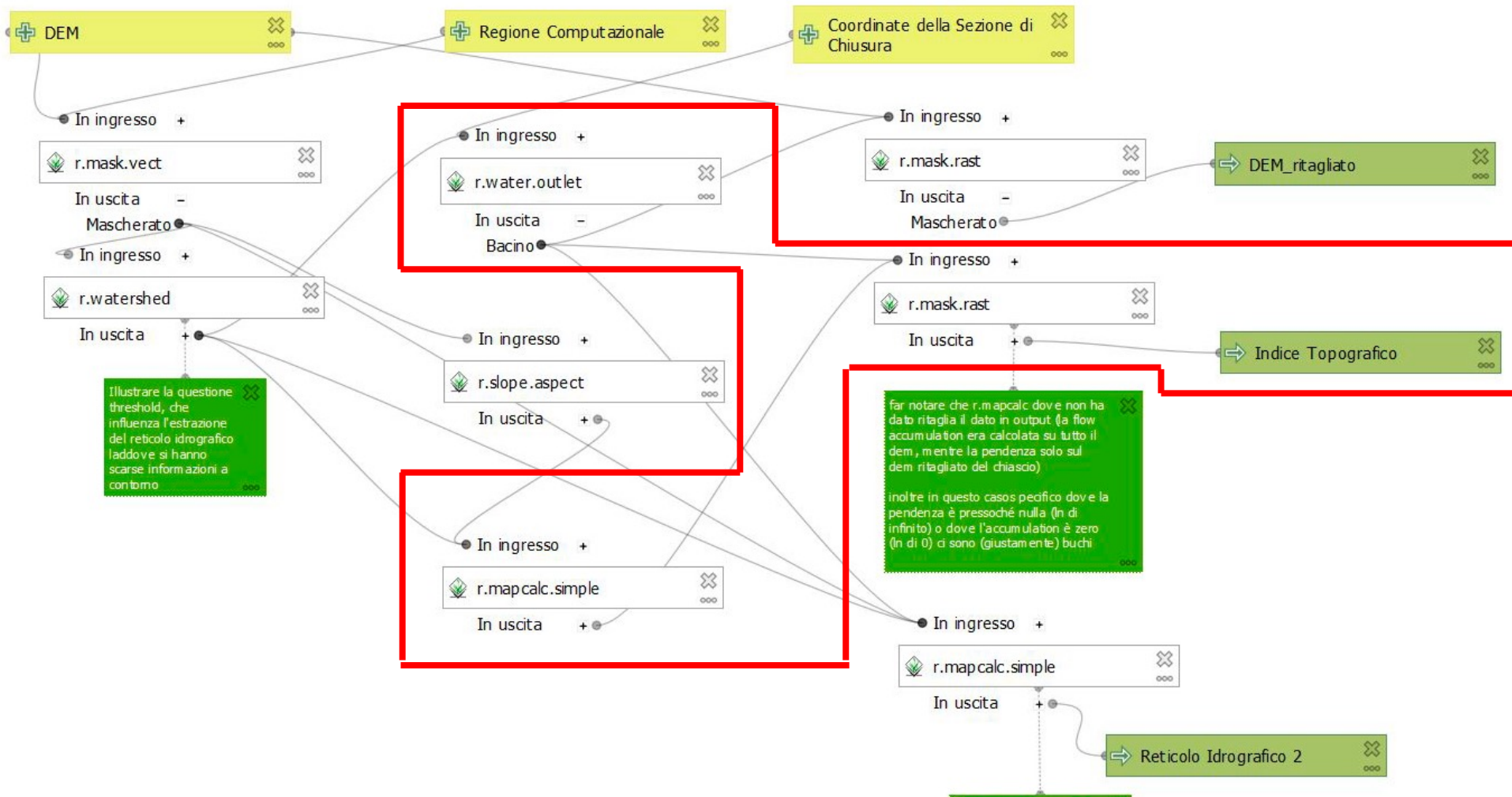
Dependencies

0 dipendenze selezionate

OK Annulla Aiuto

Il raster da usare come maschera (il bacino) lo richiamiamo come “risultato di algoritmo” (r.water.outlet) mentre il raster da mascherare sarà “modello in ingresso”, ovvero il DEM in input.

Questa volta associamo un nome all’output:
“DEM_ritagliato”



Produzione dei layers in output ritagliati: il TI

Ritagliamo ora la mappa dell'indice topografico, sempre sul bacino e sempre utilizzando r.mask.rast.

Il raster da usare come maschera è sempre il bacino, lo richiamiamo quindi come “risultato di algoritmo” (r.water.outlet) mentre il raster da mascherare è quello prodotto da r.mapcalc

sempre classificato come “risultato algoritmo”, ovvero la mappa dell'indice topografico

Anche questa volta associamo un nome all'output: “**Indice Topografico**”

r.mask.rast

Properties Comments

Description r.mask.rast

Show advanced parameters

Nome del raster da usare come maschera

Utilizzo dell'algoritmo in uscita "Bacino" dall'algoritmo "r.water.outlet"

Nome del raster a cui applicare la maschera

Utilizzo dell'algoritmo in uscita "Calcolato" dall'algoritmo "r.mapcalc.simple"

Valori raster da usare per maschera. Formato: 1 2 3 a 7 * [opzionale]

123 *

Mascherato

Indice Topografico

Dependencies

0 dipendenze selezionate

OK Annulla Aiuto

Produzione dei layers in output ritagliati: il reticolo “parlante”

Per ritagliare ora il reticolo sul bacino ed associargli una quota, da leggere dal DEM, usiamo ancora **r.mapcalc**.

Questa volta gli operandi sono 3: il bacino, il DEM (prendiamo quello ritagliato da r.mask.vect) e il reticolo.

r.mapcalc.simple

Properties Comments

Description r.mapcalc.simple

Show advanced parameters

Layer raster A
Utilizzo dell'algoritmo in uscita "Mascherato" dall'algoritmo "r.mask.vect"

Raster B [opzionale]
Utilizzo dell'algoritmo in uscita "Segmenti di corso d'acqua" dall'algoritmo "r.watershed"

Raster C [opzionale]
Utilizzo dell'algoritmo in uscita "Bacino" dall'algoritmo "r.water.outlet"

Raster D [opzionale]
123

Layer raster E [opzionale]
123

Layer raster F [opzionale]
123

Formula
123 if(C && B, A, null())

Calcolato
Reticolo Idrografico 2

OK Annulla Aiuto

Tramite formula andremo a dirgli di prendere il valore dell'elevazione (DEM) quando, sotto la medesima cella, ci sono sia il bacino che il reticolo. Altrimenti si considera un valore nullo.

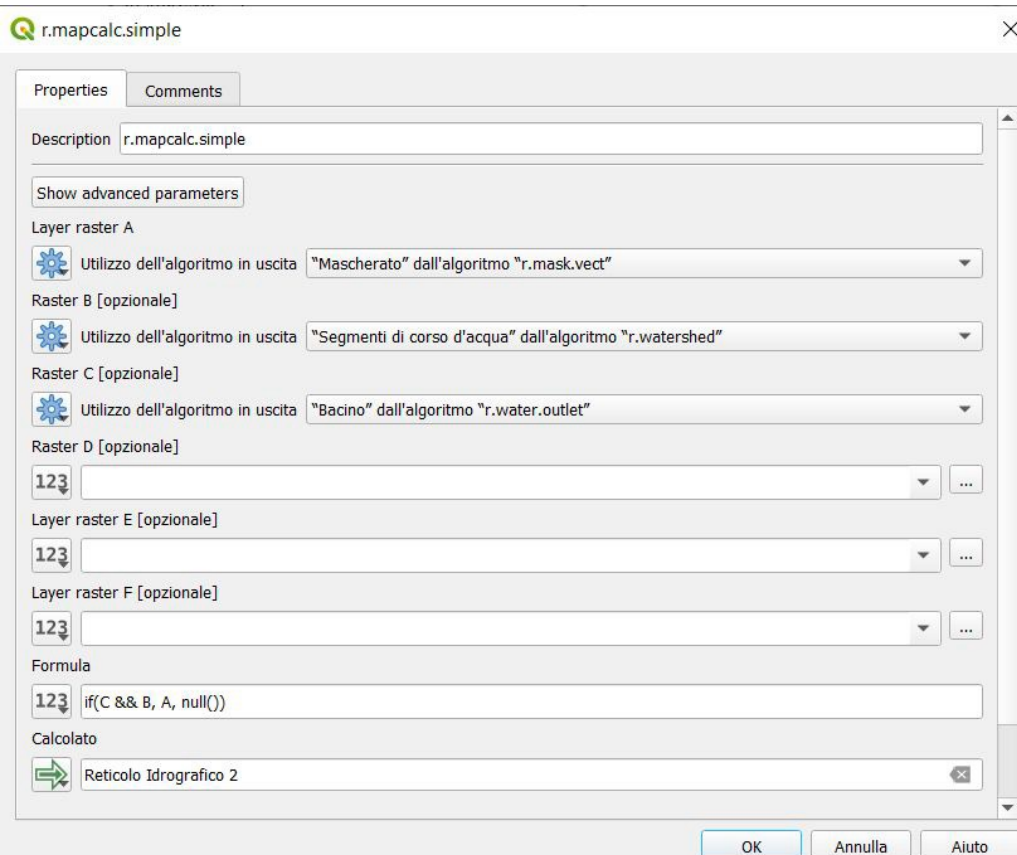
Tutti e tre i layers (DEM, reticolo e bacino) saranno quindi richiamati come “risultato di algoritmo” (da r.mask.vect, r.watershed e r.water.outlet rispettivamente) divenendo A, B e C della nostra operazione

Produzione dei layers in output ritagliati: il reticolo “parlante”

Per ritagliare ora il reticolo sul bacino ed associargli una quota, da leggere dal DEM, usiamo ancora **r.mapcalc**.

La formula utilizza operatori condizionali, logici e variabili interne ed ha sintassi:

`if(C && B, A, null())`



function	description
if(x,a,b)	a if x not zero, b otherwise

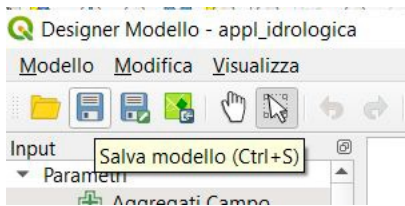
Operator	Meaning	Type	Precedence
&&	logical and	Logical	3

Internal variables:
null() NULL value

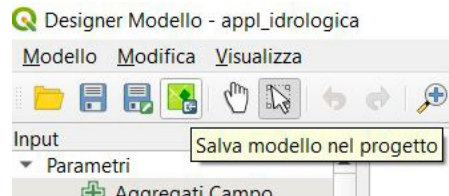
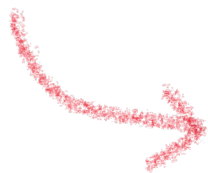
Anche in questo caso associamo un nome all'output: **“Reticolo Idrografico 2”**

Salvare il modello, inviarlo al progetto e richiamarlo come modello di progetto

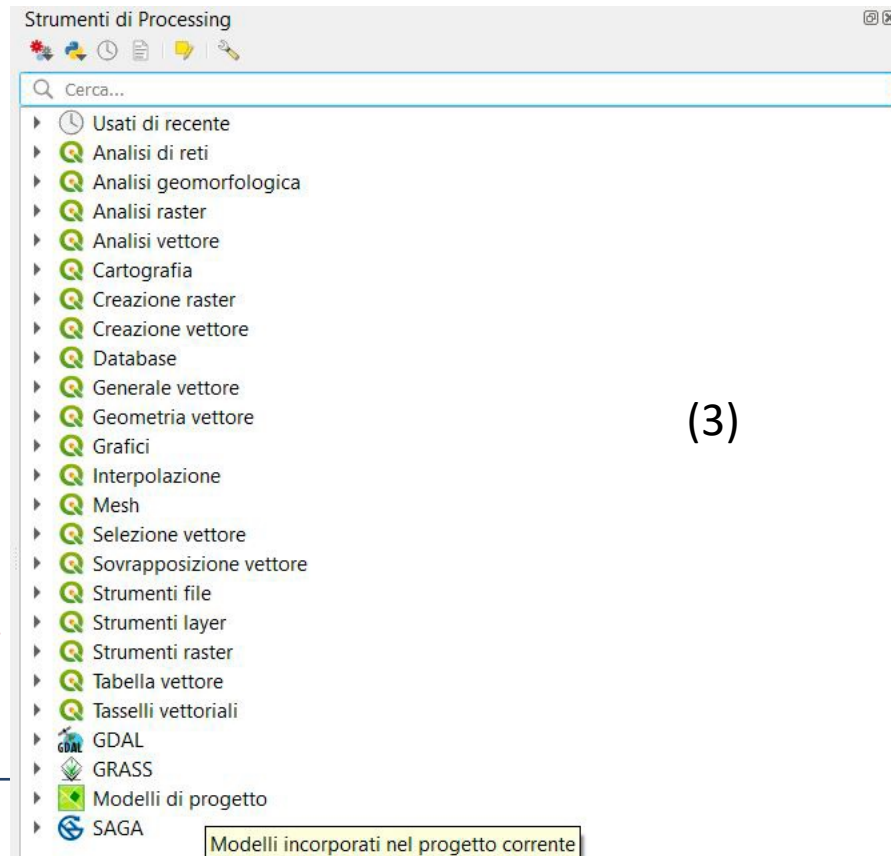
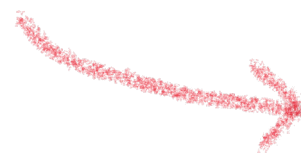
Una volta costruito il modello con il model builder sarà sufficiente salvarlo tramite l'apposito tasto (1). Si genera così un file con estensione .model3 che sarà possibile poi richiamare in seguito da ogni altra istanza di model builder. Poi possiamo passare il modello al progetto di QGIS attivo (2) e quindi richiamarlo direttamente dalla toolbox di processing (3)



(1)



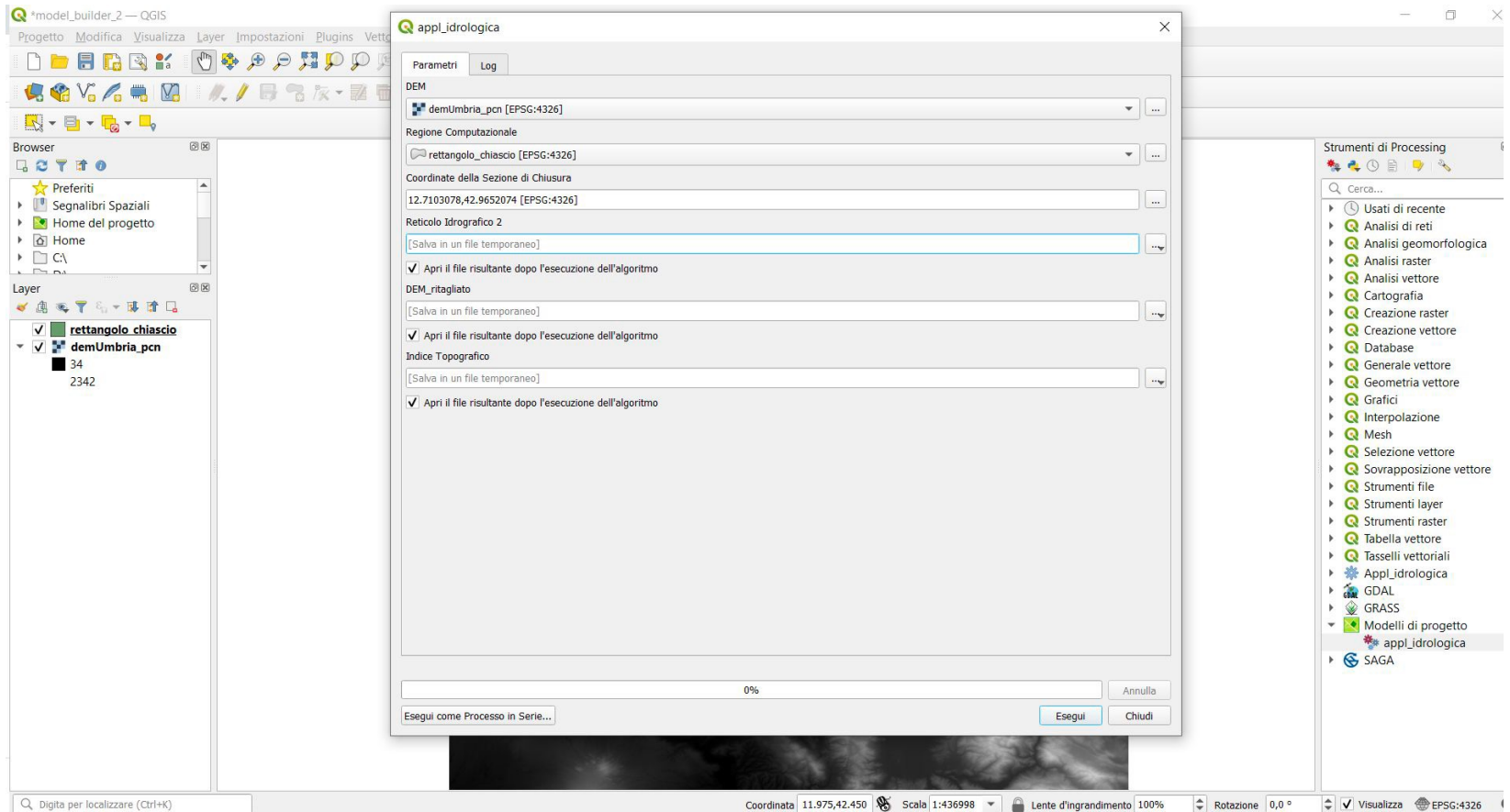
(2)



(3)

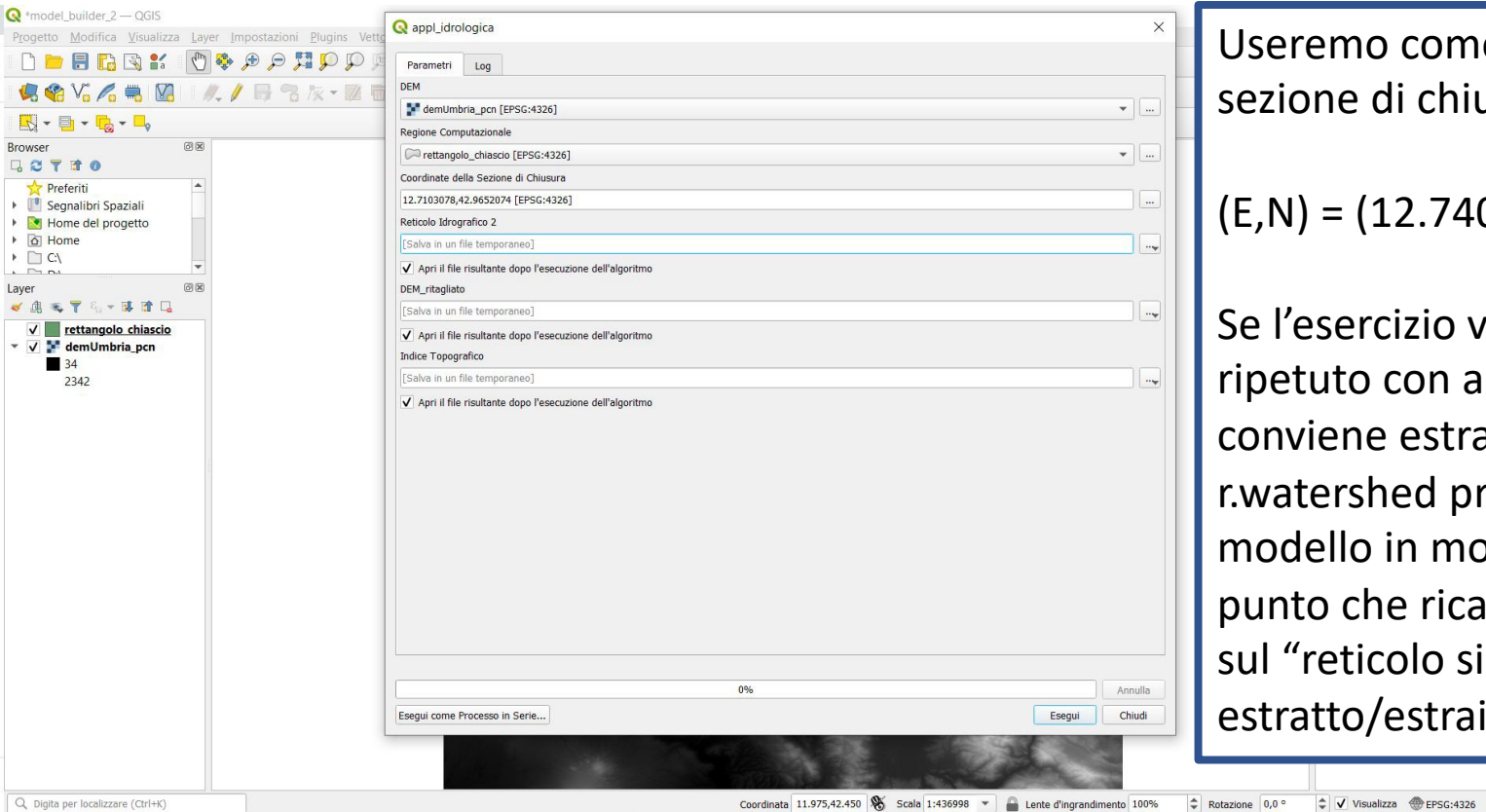
Salvare il modello, inviarlo al progetto e richiamarlo come modello di progetto

A questo punto possiamo far girare il modello dal progetto di QGIS cercandolo nella processing toolbox:



Salvare il modello, inviarlo al progetto e richiamarlo come modello di progetto

A questo punto possiamo far girare il modello dal progetto di QGIS cercandolo nella processing toolbox:



Useremo come coordinate della sezione di chiusura:

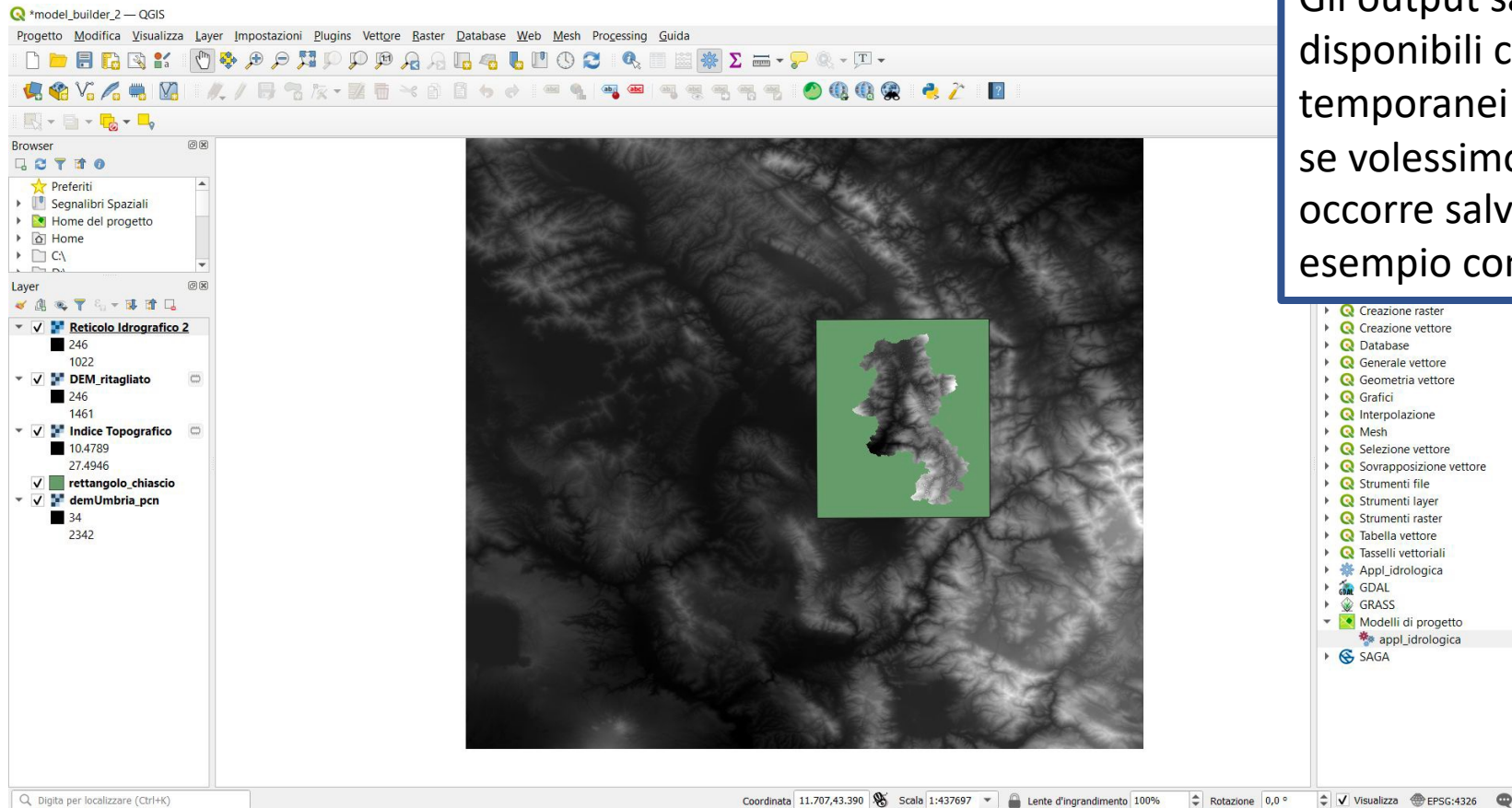
$$(E,N) = (12.7409895,43.0073769)$$

Se l'esercizio volesse essere ripetuto con altre coordinate conviene estrarre un reticolo con r.watershed prima di far girare il modello in modo da scegliere un punto che ricada effettivamente sul "reticolo sintetico" estratto/estraibile da DEM.

Salvare il modello, inviarlo al progetto e richiamarlo come modello di progetto

A questo punto possiamo far girare il modello dal progetto di QGIS cercandolo nella processing toolbox:

Gli output saranno disponibili come layers temporanei nella TOC, se volessimo archivarli occorre salvarli, ad esempio come geotiff.

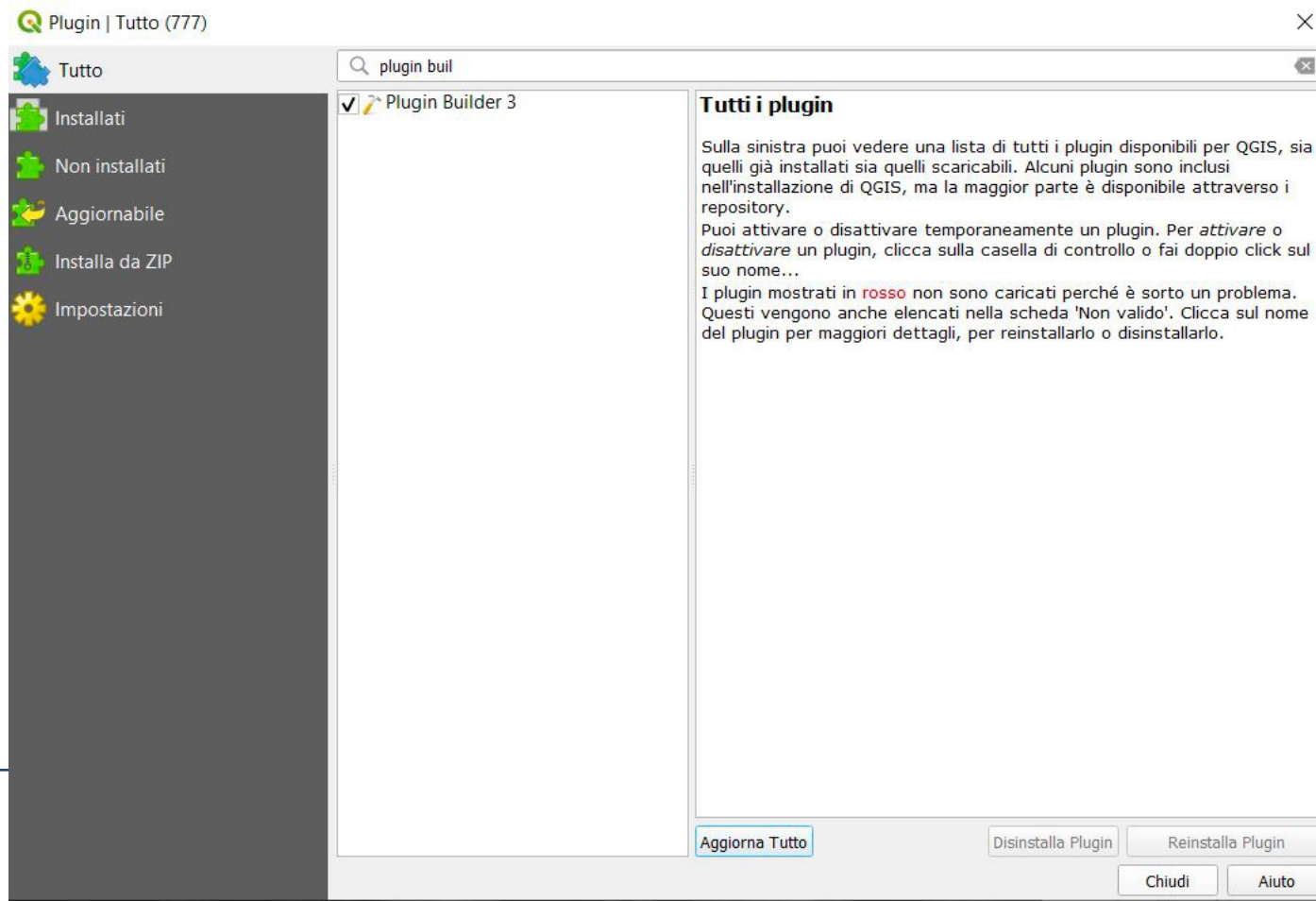


Creare un plugin per QGIS a partire dal modello

Il modello appena visto può essere anche distribuito come plugin di QGIS tramite pochi semplici passaggi:

1. Installare il plugin di QGIS “Plugin Builder” andando su Plugins > Gestisci e Installa Plugins.

Cercando «Plugin Builder» otterremo quello che cerchiamo



Creare un plugin per QGIS a partire dal modello

Il modello appena visto può essere anche distribuito come plugin di QGIS tramite pochi semplici passaggi:

2. Scegliendo “creare nuovo plugin” andiamo a creare la “struttura” del nuovo plugin, tutte le cartelle e files accessori al codice sono creati in questa fase

QGIS Plugin Builder - 3.2.1

QGIS Plugin Builder

Class name: Appl_idrologica

Plugin name: Appl_idrologica

Description: Appl_idrologica

Module name: Appl_idrologica

Version number: 0.1

Minimum QGIS version: 3.0

Author/Company: annagrass

Email address: annagrass@gmail.com

Aiuto <Previous Next > Annulla

QGIS Plugin Builder - 3.2.1

QGIS Plugin Builder

Template: Tool button with dialog, Tool button with dock widget, Processing Provider

Text for the menu item: e.g. Link photo

Menu: Plugins

Aiuto <Previous Next > Annulla

QGIS Plugin Builder - 3.2.1

QGIS Plugin Builder

Publication (mandatory Items)

Bug tracker: http://bugs

Repository: http://repo

Publication (recommended Items)

Home page: http://homepage

Tags: idrologia, python, mapalgebra

Flag the plugin as experimental

Aiuto <Previous Next > Annulla

Creare un plugin per QGIS a partire dal modello

Il modello appena visto può essere anche distribuito come plugin di QGIS tramite pochi semplici passaggi:

3. nella cartella appena creata ci interessano due files: `Appl_idrologica_agorithm.py` e `_provider.py`. In `_algorithm.py` va inserito il nostro codice al posto di quello di esempio (da riga 33 in poi)

```
C:\Users\annalisa.minelli\Documents\corsoAvanzatoQgis\per_diapo\appl_idrologica\Appl_idrologica_algorithm.py - Notepad++
File Modifica Cerca Visualizza Formato Linguaggio Configurazione Strumenti Macro Esegui Plugin Finestra ?
change log 12 Appl_idrologica_provider.py 12 Appl_idrologica_algorithm.py
1 # -*- coding: utf-8 -*-
2
3 """
4
5 Appl_idrologica
6
7 Appl_idrologica
8 Generated by Plugin Builder: http://q-sherman.github.io/Qgis-Plugin-Builder/
9
10 begin      : 2021-11-11
11 copyright  : (C) 2021 by annagrass
12 email      : annagrass@gmail.com
13
14
15
16
17 * This program is free software: you can redistribute it and/or modify
18 * it under the terms of the GNU General Public License as published by
19 * the Free Software Foundation: either version 2 of the License, or
20 * (at your option) any later version.
21 *
22 *
23 """
24
25 _author_ = 'annagrass'
26 _date_ = '2021-11-11'
27 _copyright_ = '(C) 2021 by annagrass'
28
29 # This will get replaced with a git SHA1 when you do a git archive
30
31 _revision_ = '$Format:%H$'
32
33 from qgis.core import QgsProcessing
34 from qgis.core import QgsProcessingAlgorithm
35 from qgis.core import QgsProcessingMultiStepFeedback
36 from qgis.core import QgsProcessingParameterRasterLayer
37 from qgis.core import QgsProcessingParameterVectorLayer
38 from qgis.core import QgsProcessingParameterPoint
39 from qgis.core import QgsProcessingParameterRasterDestination
40 import processing
41
42 class Appl_idrologica(QgsProcessingAlgorithm):
```

The image shows a QGIS Modeler workflow and its corresponding Python code in the Processing Script Editor. The workflow includes the following steps:

- Input: DEM, Regione Computazionale, Coordinate della Sezione di Chiusura
- Process: r.mask.vect (In ingresso), Mascherato (In ingresso), r.watershed (In ingresso)
- Output: In uscita

The Processing Script Editor shows the following code:

```
1 """
2 Model exported as python.
3 Name: : appl_idrologica
4 Group: :
5 With QGIS: : 31609
6
7
8 from qgis.core import QgsProcessing
9 from qgis.core import QgsProcessingAlgorithm
10 from qgis.core import QgsProcessingMultiStepFeedback
11 from qgis.core import QgsProcessingParameterRasterLayer
12 from qgis.core import QgsProcessingParameterVectorLayer
13 from qgis.core import QgsProcessingParameterPoint
14 from qgis.core import QgsProcessingParameterRasterDestination
15 import processing
16
17
18 class Appl_idrologica(QgsProcessingAlgorithm):
19
20     def __initAlgorithm(self, config=None):
21         self.addParameter(QgsProcessingParameterRasterLayer('DEM', 'DEM'))
22         self.addParameter(QgsProcessingParameterVectorLayer('Regione Computazionale', 'Regione Computazionale'))
23         self.addParameter(QgsProcessingParameterPoint('Coordinate della Sezione di Chiusura', 'Coordinate della Sezione di Chiusura'))
24
```

Creare un plugin per QGIS a partire dal modello

Il modello appena visto può essere anche distribuito come plugin di QGIS tramite pochi semplici passaggi:

3. nella cartella appena creata ci interessano due files: `Appl_idrologica_agorithm.py` e `_provider.py`. In `_algorithm.py` va inserito il nostro codice al posto di quello di esempio (da riga 33 in poi)

The image displays a development environment for creating a QGIS plugin. On the left, a code editor shows the `Appl_idrologicaProvider` class in `_provider.py`. The `loadAlgorithms` method (lines 52-58) is highlighted, showing the addition of the `Appl_idrologica` algorithm. A blue arrow points from this code to the 'Designer Modello' window in the center. This window shows a Processing Model with inputs: 'DEM', 'Regione Computazionale', and 'Coordinate della Sezione di Chiusura'. The model steps include 'r.mask.vect', 'Mascherato', 'r.watershed', and 'In uscita'. On the right, the 'Processing Script Editor' shows the Python code for the `Appl_idrologica` algorithm, starting with imports and the `initAlgorithm` method (lines 18-24).

Creare un plugin per QGIS a partire dal modello

Il modello appena visto può essere anche distribuito come plugin di QGIS tramite pochi semplici passaggi:

- ora non ci resta che fare uno zip della cartella Appl_idrologica e richiamarlo dall'installatore di plugins. Una volta installato il plugin esso apparirà nella processing toolbox di QGIS

The image shows two overlapping windows from the QGIS interface. The background window is the 'Plugin | Installa da ZIP' dialog. It displays instructions for installing a plugin from a ZIP file and shows the file path: 'C:\Users\annalisa.minelli\Documents\corsoAvanzatoQgis\per_diapo\appl_idrologica.zip'. An 'Installa Plugin' button is visible at the bottom.

The foreground window is the 'appl_idrologica' parameter dialog. It has a 'Parametri' tab and a 'Log' tab. The 'DEM' section is expanded, showing a dropdown menu set to 'Reticolo Idrografico 2 [EPSG:4326]'. Below this, there are several sections with checkboxes and file paths, all of which are checked: 'Apri il file risultante dopo l'esecuzione dell'algoritmo' for 'DEM_ritagliato', 'Indice Topografico', and another unnamed section. At the bottom, there is a progress bar at 0% and buttons for 'Esegui' and 'Chiudi'.

On the right side, the 'Strumenti di Processing' toolbox is visible, showing a list of processing tools. The 'appl_idrologica' plugin is listed at the bottom of the toolbox, indicating it has been successfully installed.

FINE: Grazie per l'attenzione!



Dubbi? Perplexità?

annalisa.minelli@isprambiente.it