Robert Pfeffer

Maulwurfshügel in QGIS

Reliefdarstellung in Maulwurfshügelmanier auf der Grundlage digitaler Höhendaten

‚Maulwurfshügel' sind eine altertümliche Art der Reliefdarstellung, die verbreitet in Karten des 16. bis 18. Jahrhunderts begegnet sowie, inspiriert hiervon, in Karten des Fantasy-Genres. Maulwurfshügel bieten sich an, wo eine bestimmte antike Ästhetik gewünscht ist, während geografische Genauigkeit nicht im Vordergrund steht. Dieser Abriss erläutert, wie sich als SVG-Grafiken vorbereitete Hügel anhand heruntergeladener Höhendaten mit etwas zusätzlicher Arbeit in QGIS automatisch auf der Karte anordnen lassen.[1]

Das geschieht im Wesentlichen in sechs Schritten:

I. Herunterladen der Höhendaten,

II. Umwandeln der Höhenkarte in eine Steilheitskarte,

III. Ableiten von Mittel- und Hochgebirgsflächen aus der Steilheitskarte,

IV. Anlegen von Quellebenen und Zielebene für die Hügelpunkte,

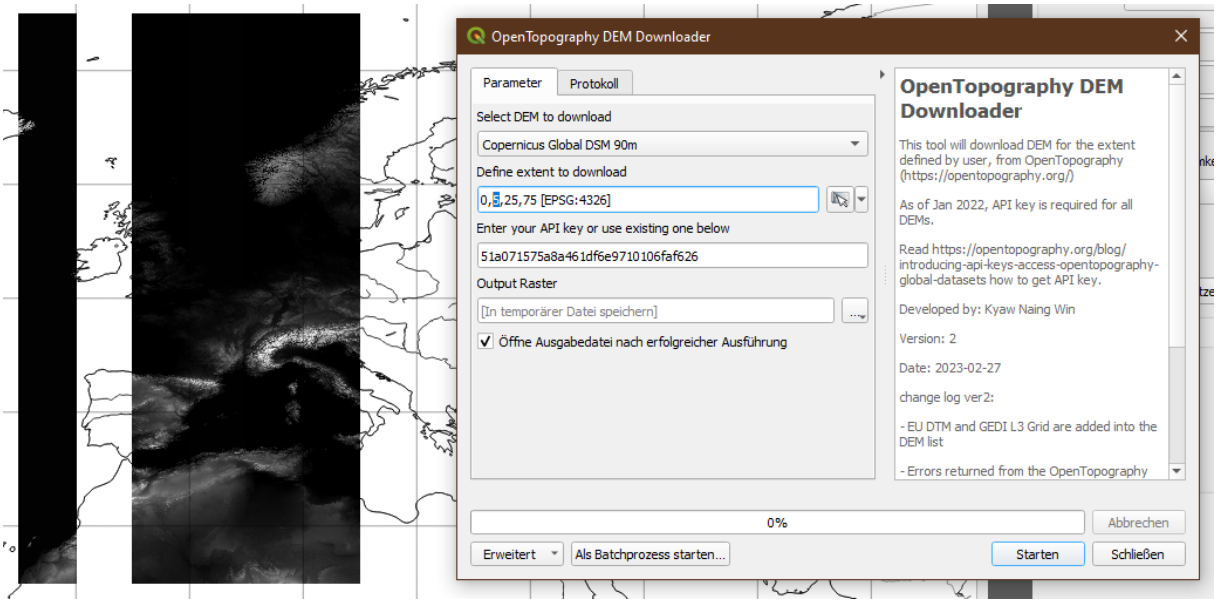
V. Generieren der Punkte für verschiedene Maßstäbe und

VI. Darstellung der Punkte mit Hügelgrafiken.

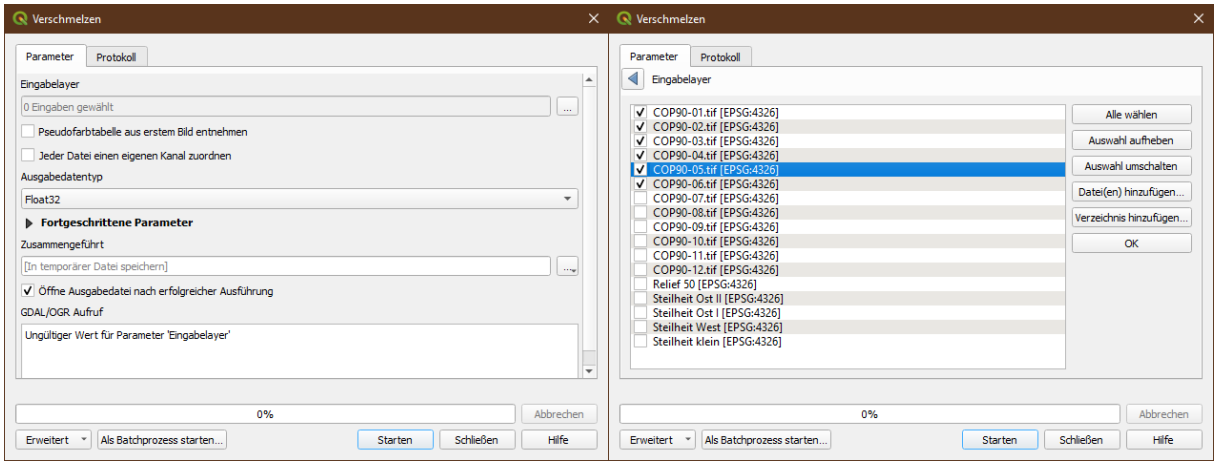
[1] Getestet wurde es hier in QGIS 3.28 und 3.34. — Die hier verwendeten Python-Skripte habe ich nicht allein programmiert, da ich darin nicht bewandert bin. ChatGPT war mir eine große Hilfe!

I. Herunterladen der Höhendaten

Zunächst wird eine Höhenkarte für das betreffende Gebiet benötigt. Dafür können wir das Plugin *OpenTopography DEM Downloader* verwenden und z.B. Copernicus-90-Daten herunterladen. Da der Server stets nur begrenzte Volumina herausrückt, müssen sie ggf. gestückelt heruntergeladen werden; im nachstehenden Beispiel geschieht dies in 5° breiten Streifen:



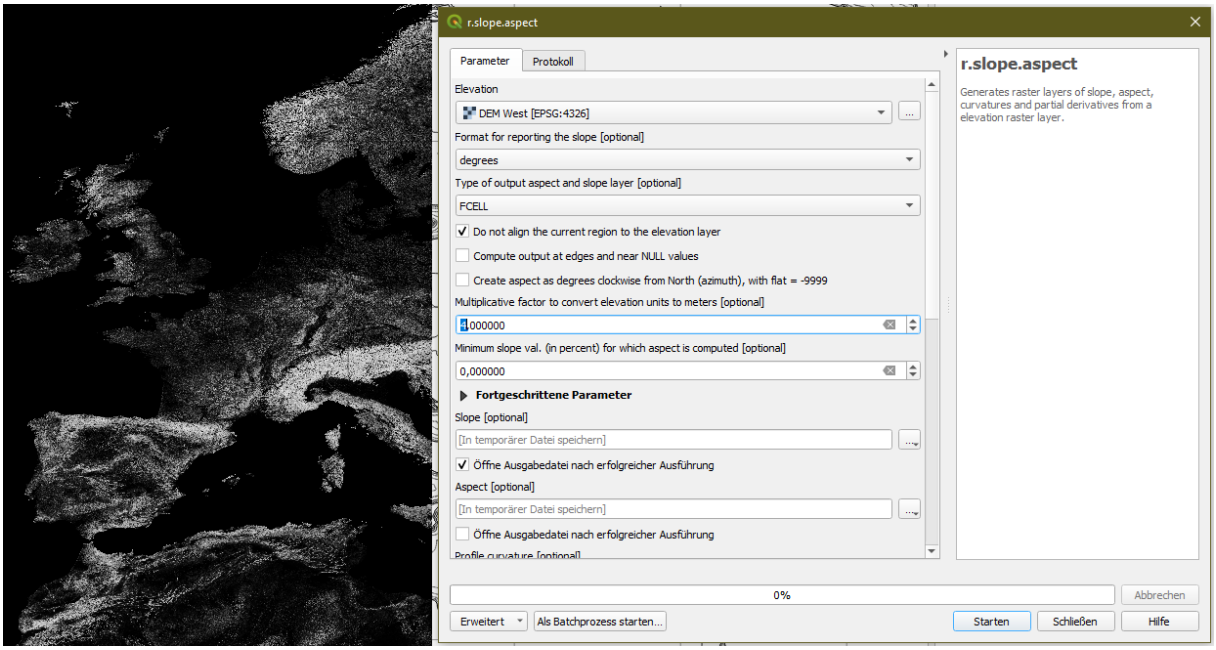
Die vielen einzelnen Ebenen, die dabei entstehen, sollten vor ihrer weiteren Verarbeitung zu einer Ebene vereinigt werden. Wenn QGIS dabei wegen der hohen Datenmengen an seine Leistungsgrenzen stößt, muss man sich ggf. mit mehreren, möglichst wenigen Ebenen begnügen. Die Funktion zum Verschmelzen findet sich im Menü unter *Raster —> Sonstiges —> Verschmelzen:*



II. Umwandeln der Höhenkarte in eine Steilheitskarte

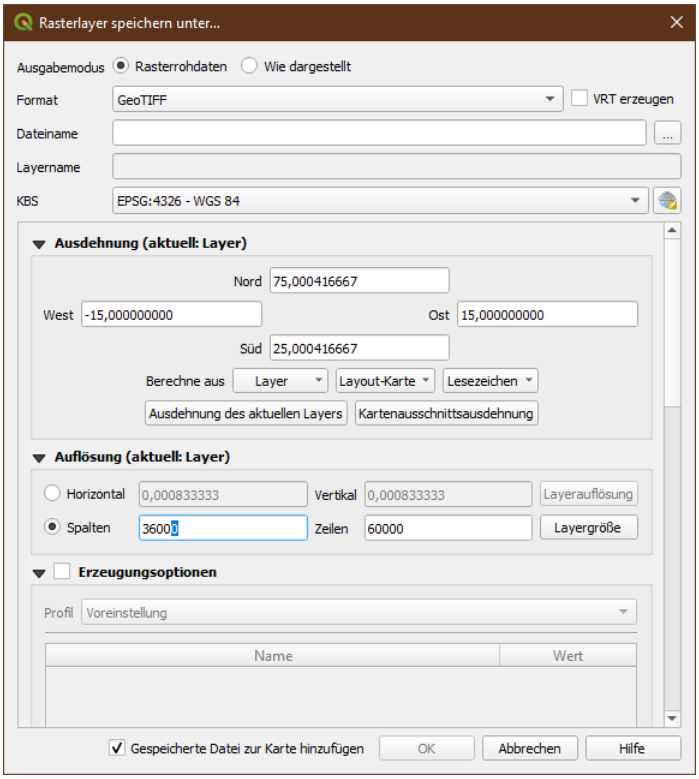
Nun haben wir eine schöne Höhenkarte, die aber noch nicht viel nützt – denn wir wollen ja keine Ebene mit Hügeln besäen, nur weil es eine Hochebene ist, während wir umgekehrt Küstengebirge auch dann abbilden wollen, wenn sie nicht weit über Null hinausragen. Was wir stattdessen brauchen, ist also eine Karte, die nicht die Höhe darstellt, sondern die Steilheit des Geländes.

Dafür bemühen wir die GRASS-GIS-Funktion *r.slope.aspect*. In deren Fenster muss unter *Fortgeschrittene Parameter* nur das Häkchen hinter ‚Slope' bleiben, alle weiteren können weg. Als Multiplikator habe ich 4 gewählt; gewiss gehen auch andere Werte.



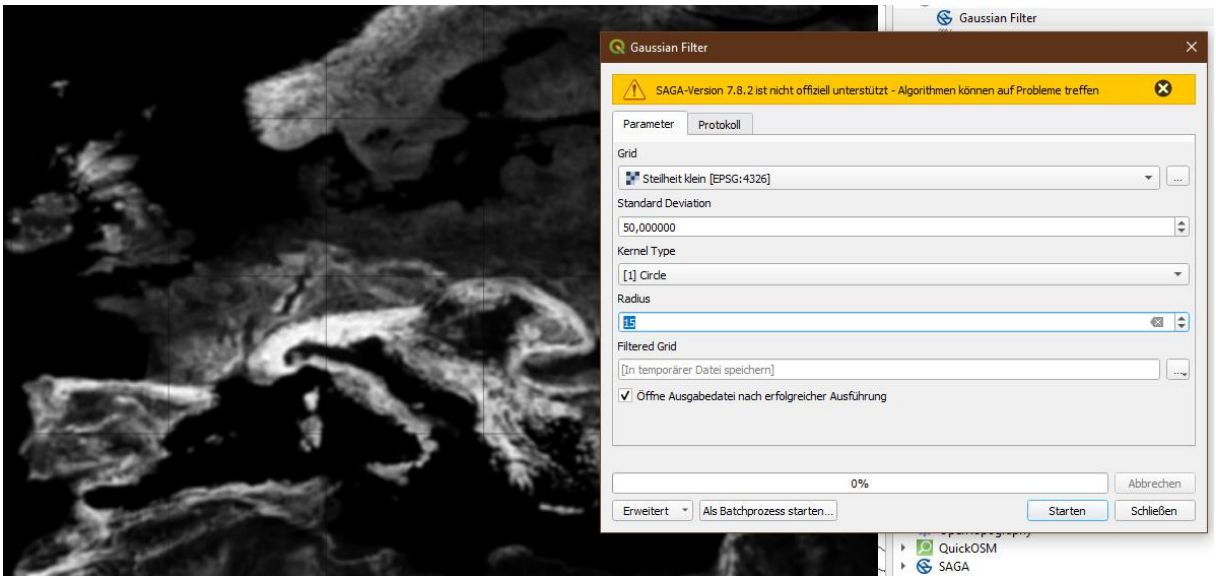
Auf diese Weise entsteht eine Karte, die nicht mehr zeigt, wie hoch gelegen, sondern wie steil zerklüftet das Gelände ist. Man beachte etwa die Hochebenen Spaniens und Marokkos: Sie sind nicht mehr grau (weil hoch), sondern schwarz (weil eben).

Um die weitere Verarbeitung zu erleichtern, verkleinern wir die Grafik erst einmal auf ein Zwanzigstel, indem wir nach Rechtsklick auf die Ebene —> *Export* —> *Speichern als* … wählen. Im nun geöffneten Fenster entfernen wir bei Spalten und Zeilen je eine Null und teilen den Rest durch zwei, wählen unter *Dateiname* einen Speicherort und speichern die Grafik als neue Ebene (s. rechts).



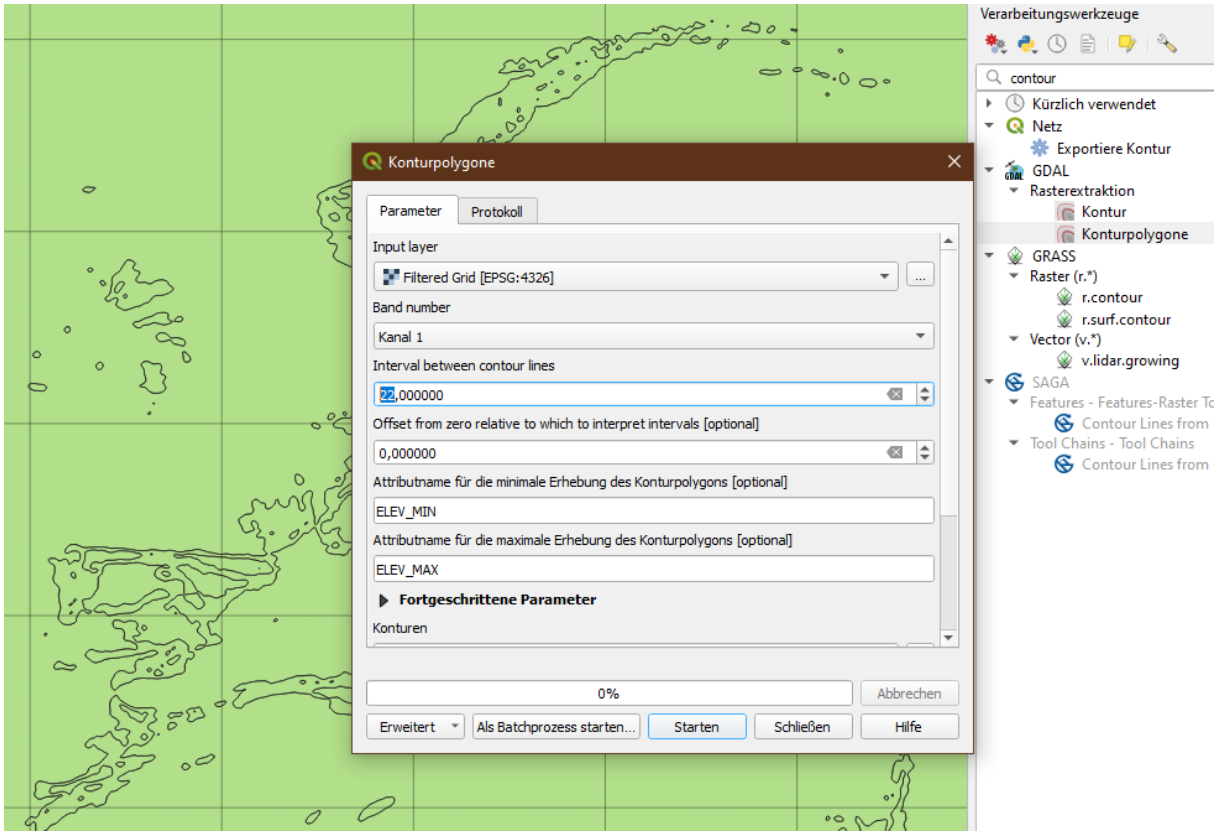
Diese verkleinerte Karte ist allerdings zur weiteren Verwendung, d.h. Vektorisierung, noch immer zu feinkörnig. Wir belegen sie darum mit einer Gauß'schen Unschärfe, indem wir aus den SAGA-Werkzeugen den *Gaussian Filter* anwenden[2]; im hiesigen Beispiel schien dabei ein Radius von 15 angemessen (s. folgende Seite).

[2] QGIS 3.28 scheint die letzte Version zu sein, die SAGA-Werkzeuge noch von Haus aus unterstützt. In QGIS 3.34 ist dafür schon das Plugin „Processing Saga NextGen Provider" und seine Einstellung unter *Einstellungen –> Optionen –> Verarbeitung –> Datenanbieter* notwendig.



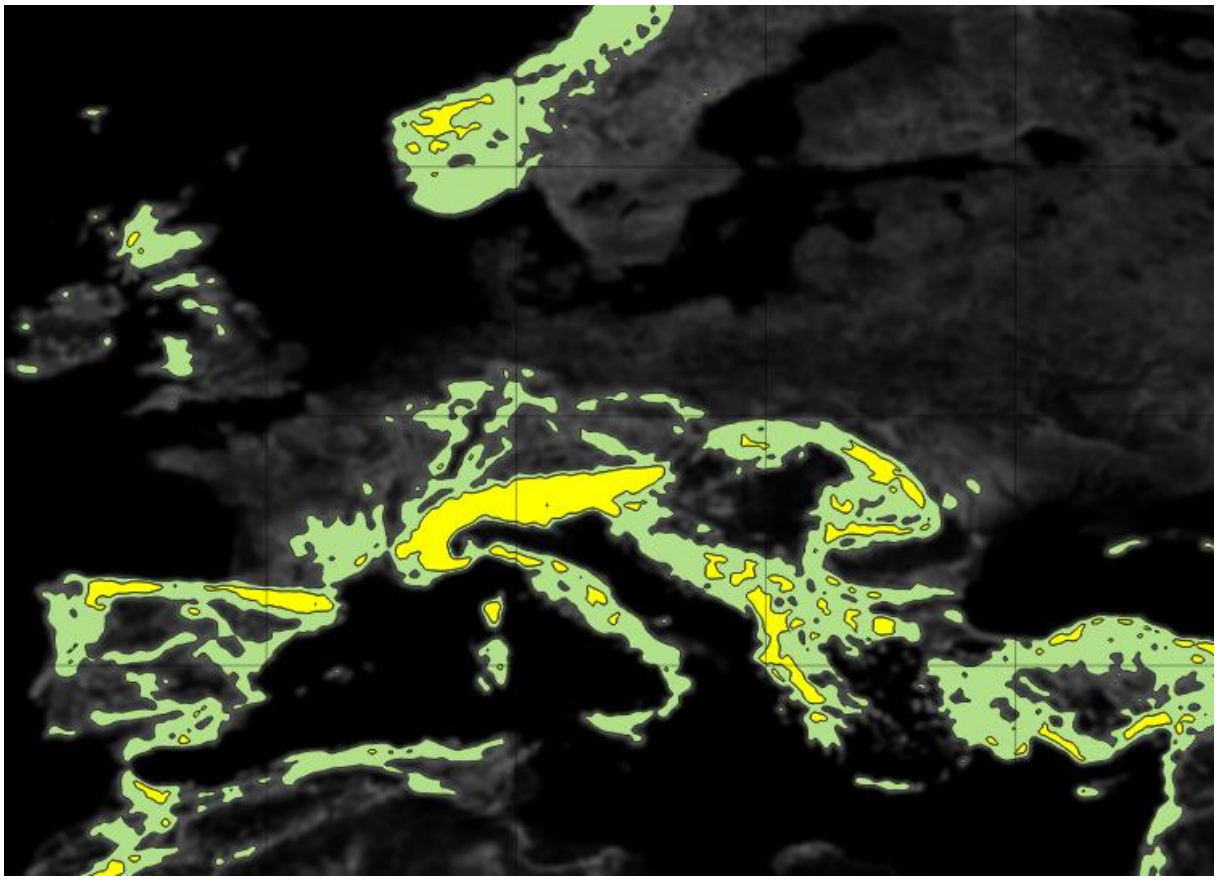
III. Ableiten von Mittel- und Hochgebirgsflächen aus der Steilheitskarte

Nun müssen die von Mittel- und Hochgebirge bedeckten Flächen gefunden und als Polygone extrahiert werden. Dafür nutzen wir die GDAL-Funktion *Konturpolygone,* die eigentlich für Höhenlinien gedacht ist. Im vorliegenden Beispiel führt ein Intervall-Wert von 22 zu passenden Ergebnissen:



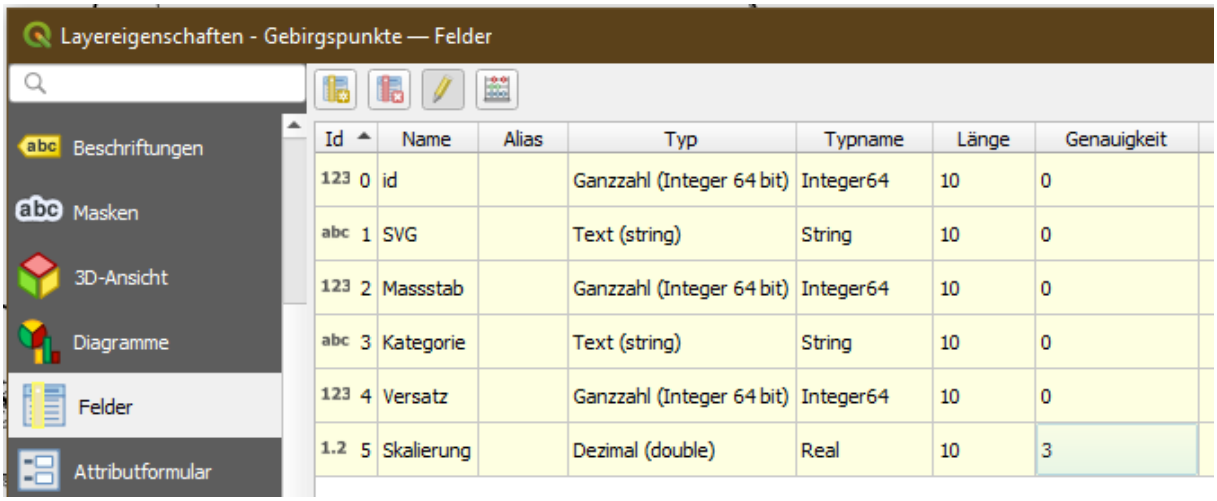
Auf der neu entstandenen Ebene haben wir jetzt drei Polygone: Eines erstreckt sich bis zu den Kartenrändern und über alle relativ ebenen Flächen einschließlich der Meere; dieses können wir löschen. Übrig bleiben die zwei Polygone für Mittelgebirgs- und Hochgebirgsflächen. Sollte es noch ein weiteres für die steilsten Gipfel geben, können wir es ebenfalls löschen, falls es

vernachlässigbar klein ist, oder wir starten die Prozedur mit einem höheren Wert als 22 erneut. (Gibt es gar noch mehr Polygone, war der Wert ohnehin zu niedrig.)



IV. Anlegen von Quellebenen und Zielebene für die Hügelpunkte

Die beiden verbleibenden Polygone verteilen wir auf zwei Ebenen, die wir „Mittelgebirge“ und „Hochgebirge“ nennen. Zusätzlich erstellen wir eine weitere Vektorebene, deren Geometrie in Punkten besteht und die wir „Gebirgspunkte“ nennen. (Diese Ebenen kann man natürlich auch anders nennen, nur muss man dann die Zufallspunkte-Funktionen, die sogleich erläutert werden, an den entsprechenden Stellen ebenfalls ändern, s. Anhang.) Der Ebene „Gebirgspunkte“ müssen – entweder gleich bei ihrer Erstellung oder später in den Layer-Eigenschaften – die folgenden Felder hinzugefügt werden (außer „id“, welches schon besteht, aber nutzlos ist):



Id	Name	Alias	Typ	Typname	Länge	Genauigkeit
123 0	id		Ganzzahl (Integer 64 bit)	Integer64	10	0
abc 1	SVG		Text (string)	String	10	0
123 2	Massstab		Ganzzahl (Integer 64 bit)	Integer64	10	0
abc 3	Kategorie		Text (string)	String	10	0
123 4	Versatz		Ganzzahl (Integer 64 bit)	Integer64	10	0
1.2 5	Skalierung		Dezimal (double)	Real	10	3

Im Folgenden wird es darum gehen, die Flächen der zwei Polygone mit zufällig angeordneten Punkten zu füllen, die wiederum von zufällig gewählten SVG-Grafiken dargestellt werden. Die beiden erstgenannten Ebenen dienen dabei als Quell-Ebenen, aus denen heraus die Punkte generiert werden, und die dritte ist die Ziel-Ebene, in der sie abgespeichert werden.

Dies geschieht mittels einer Python-Funktion[3], welche in zwei Varianten vorliegt – einmal für die Mittel- und einmal für die Hochgebirge (s. Anhang) – und die im Wesentlichen Folgendes tut:

1. über die Gesamtausdehnung des Polygons ein Gitter spannen,
2. in alle Kästchen des Gitters je einen Punkt zufällig platzieren,
3. alle Punkte wieder löschen, die dabei nicht innerhalb des Polygons landen,
4. alle Punkte wieder löschen, die zu nah an der Geometrie bestimmter anderer Ebenen liegen, von der sie sich fernhalten sollen (z.B. Flüsse, Küstenlinien, …),
5. allen Punkten, die auch dann noch zu nah rechts oder links an jener Geometrie liegen, einen Attributwert von +1 oder -1 zuweisen, mit dem sie später davon abgerückt werden können (zu Sinn und Unsinn dieser Doppelung sogleich),
6. jedem Punkt zufällig einen Dateinamen von „MH01.svg“ bis „MH63.svg“ zuweisen (das sind die 63 Maulwurfshügel-Grafiken aus dem beiliegenden, vorbereiteten Satz) und
7. jedem Punkt einen zufälligen Skalierungs-Faktor von 0,9 bis 1,1 zuweisen, um zusätzlich etwas Abwechslung ins Kartenbild zu bekommen.

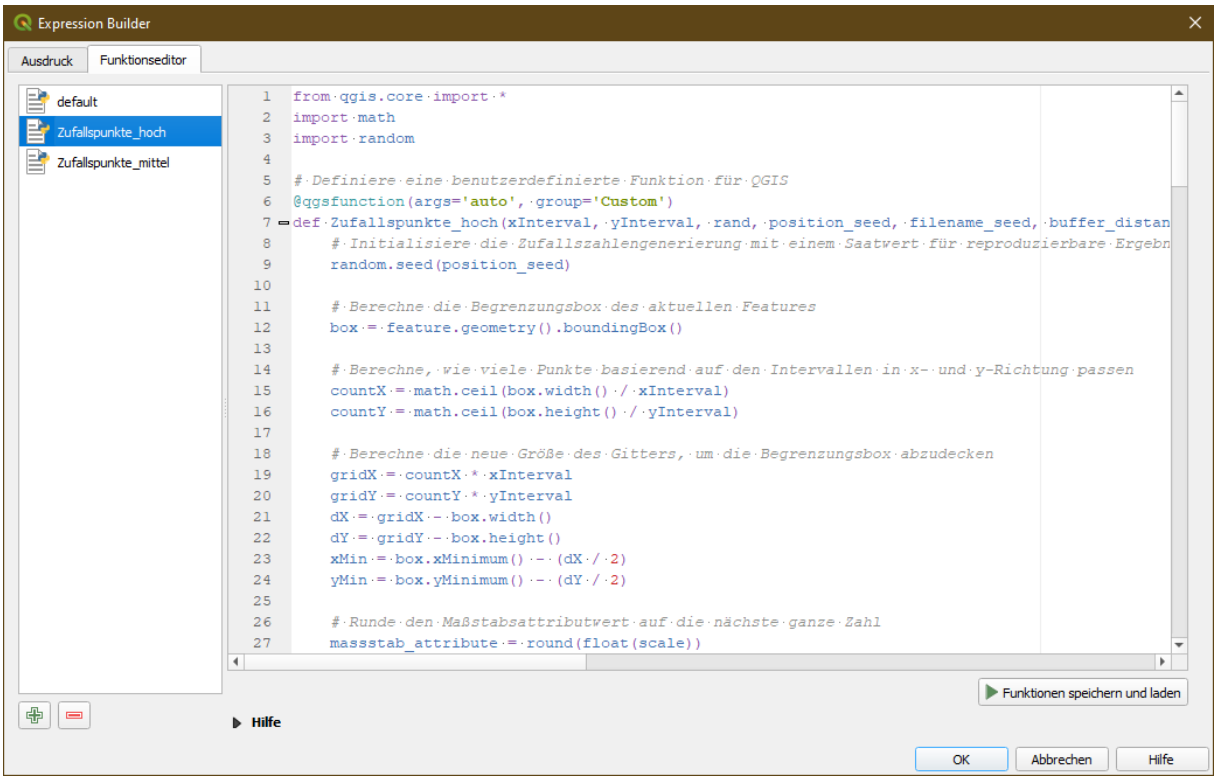
Punkte 4 und 5 verfolgen beide den Zweck, die Hügel nicht zu nah an die Geometrie bestimmter anderer Ebenen heranrücken zu lassen (die dafür namentlich in die Funktion eingetragen werden müssen, s. Anhang). In Punkt 4 wird um jene Geometrie ein Puffer gezogen, um sodann alle Punkte zu löschen, die darinnen liegen. Da unsere Hügel mehr breit als hoch sind, wäre es eigentlich schön, einen Puffer zu haben, der ebenfalls mehr breit als hoch ist. Das scheint in QGIS aber nicht vorgesehen zu sein. Mehr noch: Wenn, wie hier, Grad als Karteneinheiten dienen, sind diese in europäischen Breiten ausgerechnet mehr hoch als breit, und so gerät der Puffer genau umgekehrt, als er eigentlich sollte. Mir fiel hierfür keine andere Lösung ein, als in einem weiteren Schritt 5 noch einmal zu prüfen, welche Punkte denn dann noch im Puffer lägen, wenn sie etwas weiter rechts oder links lägen (und zwar um eine halbe Pufferbreite). Für diese Punkte wird dann ein positiver oder negativer Attributwert eingetragen, der später als Anknüpfungspunkt für einen seitlichen Versatz des Markers dienen kann[4]. (Einen Vorteil hat diese Methode: Punkte, die andernfalls gelöscht würden, bleiben erhalten und werden nur ein bisschen zu den anderen geschoben.)

In der Layergestaltung beider Quellebenen (Hochgebirge und Mittelgebirge) wählen wir nun als Füllung *Geometriegenerator,* als Geometrietyp *Punkte* und als Einheiten – im vorliegenden Beispiel – *Millimeter*. Neben dem Ausdrucksfeld drücken wir den ε-Knopf, um den Ausdrucksdialog zu öffnen. Dort wählen wir zunächst den Funktionseditor, erstellen zwei neue Funktions-Dateien namens „Zufallspunkte hoch“ und „Zufallspunkte mittel“ (oder beliebigen anderen Namen) und kopieren den Code der zwei Funktionen für Hoch- und Mittelgebirge hinein (s. Anhang). Dies können wir für beide Ebenen in einem erledigen – der Inhalt des Funktionseditors ist in beiden Ebenen verfügbar.

---

[3] Der ‚historische Kern‘ dieser Funktion ist ein Code, den Rob Jones auf https://impermanent.io/2017/05/ 05/generative-pseudo-random-polygon-fill-patterns-in-qgis/ veröffentlicht hat (zuletzt abgerufen am 02.02.24). Ihm sei an dieser Stelle dafür gedankt.

[4] S. dazu unten VI.4.



Nun wechseln wir zum Reiter *Ausdruck*, und nun wird es ebenenspezifisch: Hier kommt der Ausdruck hinein, der die Funktion aufruft, sobald die Ebene sichtbar geschaltet wird. Fürs Erste mögen folgende Ausdrücke genügen, die ein ansprechendes Punktebild im Maßstab 1 : 15 000 000 erzeugen:

Zufallspunkte_hoch(0.7, 0.35, 0.35, 33, 42, 0.25, $scale)

Zufallspunkte_mittel(0.53, 0.3, 0.48, 33, 42, 0.25, $scale)

Man beachte hierbei, dass sich „Zufallspunkte_hoch“ und „_mittel“ nicht auf die soeben vergebenen Dateinamen beziehen, sondern auf die im Code festgelegten Funktionsnamen (s. Anhang). Die Parameter in den Klammern haben im oberen der beiden Beispiele folgende Bewandtnis (s. dazu auch die Kommentierung zu Beginn beider Funktionen im Anhang):

– „0.7“ und „0.35“ sind Breite und Höhe der einzelnen Gitter-Kästchen, gemessen in Karteneinheiten, hier also Grad. Nimmt man für die Breite das Doppelte der Höhe, ergibt dies in europäischen Breiten ein ungefähr quadratisches Kästchen.

– Nochmals „0.35“ bestimmt das Maß der Zufallsstreuung, mit welcher der einzelne Punkt im jeweiligen Kästchen platziert wird.

– „33“ und „42“ sind beliebig wählbare Werte als Zufallssaaten für die Punktestreuung und für die Zuweisung der SVG-Dateinamen. So lange diese „Saat“ gleich bleibt, bleibt auch das Zufallsbild gleich. Wenn einem also das Hügelbild nicht gefällt, kann man an diesen Stellen variieren, ohne dass sich die Dichte der Punkte grundsätzlich ändert.

– „0.25“ bestimmt die Breite des Puffers, der um die Geometrie der erwähnten anderen Ebenen gezogen wird, mithin die von Punkten freizuhaltende Zone.

– Als „$scale“ wird der Kehrwert des aktuellen Maßstabs übergeben (beim Maßstab 1 : 15 000 000 also die „15 000 000“). Damit können die o.g. Ausdrücke später so verfeinert werden, dass sie Rücksicht auf den jeweiligen Maßstab nehmen (s. im Folgenden).

V. Generieren der Punkte für verschiedene Maßstäbe

Das Erzeugen der Punkte geschieht nun schlicht dadurch, dass man eine der Quellebenen sichtbar schaltet und damit die Funktion von QGIS ausführen lässt. Das kann eine Weile dauern, und es kann gut sein, dass QGIS dabei abstürzt! Mit etwas Glück sind die Punkte dann aber schon generiert und abgespeichert.

Das Generieren erfolgt zwar aus den Quellebenen heraus mittels der dort hinterlegten Funktionen, doch abgespeichert werden die Punkte in der Zielebene „Gebirgspunkte“. Die Quellebenen müssen und sollten also nur zum Zwecke der Punkteerzeugung sichtbar geschaltet und dann gleich wieder unsichtbar gemacht werden.

Beide Funktionen sind so programmiert, dass sie nur dann Punkte erzeugen, wenn für den aktuellen Maßstab noch keine vorhanden sind. Sie sind auch so voreingestellt, dass nicht für jeden beliebigen, krummen Maßstab Punkte erzeugt werden, sondern nur für Maßstäbe, deren Kehrwert durch 250 000 teilbar ist, was z.B. auf den hier gewählten Ausgangsmaßstab von 1 : 15 000 000 und viele seiner Halbierungen zutrifft[5]. (Das lässt sich natürlich bei Bedarf in den Funktionen ändern, s. Anhang).

Man erzeugt also die Punkte für verschiedene Maßstäbe, indem man in verschiedenen Maßstäben die Quellebene sichtbar macht oder indem man bei sichtbarer Quellebene durch die verschiedenen Maßstäbe scrollt. Dabei zeigt sich sofort das Problem, dass die als Parameter übergebenen Abstände absolut sind, so dass die Gitterkästchen und damit auch die Punkte in jedem Maßstab denselben geografischen Abstand haben. Will man die Punkte bei doppeltem Maßstab doppelt so dicht haben, damit der optische Abstand der Gleiche bleibt, darf darum bei doppeltem Maßstab nur der halbe Wert als Parameter übergeben werden. Das erreicht man, indem man statt eines absoluten Werts einen Bruch unter Verwendung von „$scale“ im Zähler einsetzt. (Zu genau diesem Zweck wird der Funktion „$scale“ als letzter Parameter übergeben.) Wenn einem also z.B. im Maßstab 1 : 15 000 000 ein horizontaler Abstand von 0,7 passend erschien – also so:

Zufallspunkte_hoch(**0.7,** …,

dann kann man diese 0,7 durch einen Bruch ersetzen, der im genannten Maßstab dasselbe ergibt wie 0,7 – nämlich 15 000 000 / 21 428 571,4 – und der unter Verwendung von „$scale“ statt „15 000 000“ sowie ein bisschen gerundet so lautet:

Zufallspunkte_hoch(**$scale/21400000,** ….

Bei doppeltem Maßstab ist dann „$scale“ nur noch halb so groß, womit der geografische Abstand halbiert wäre, so dass er optisch der Gleiche bleibt. Will man nun, dass bei doppeltem Maßstab der optische Abstand doch nicht ganz gleich bleibt, sondern doch wieder ein klein bisschen größer wird, so kann man den Parameter noch um einen geringen Summanden ergänzen, der genau umgekehrt wirkt, z.B.:

Zufallspunkte_hoch($scale/21400000+**1200000/$scale,** ….

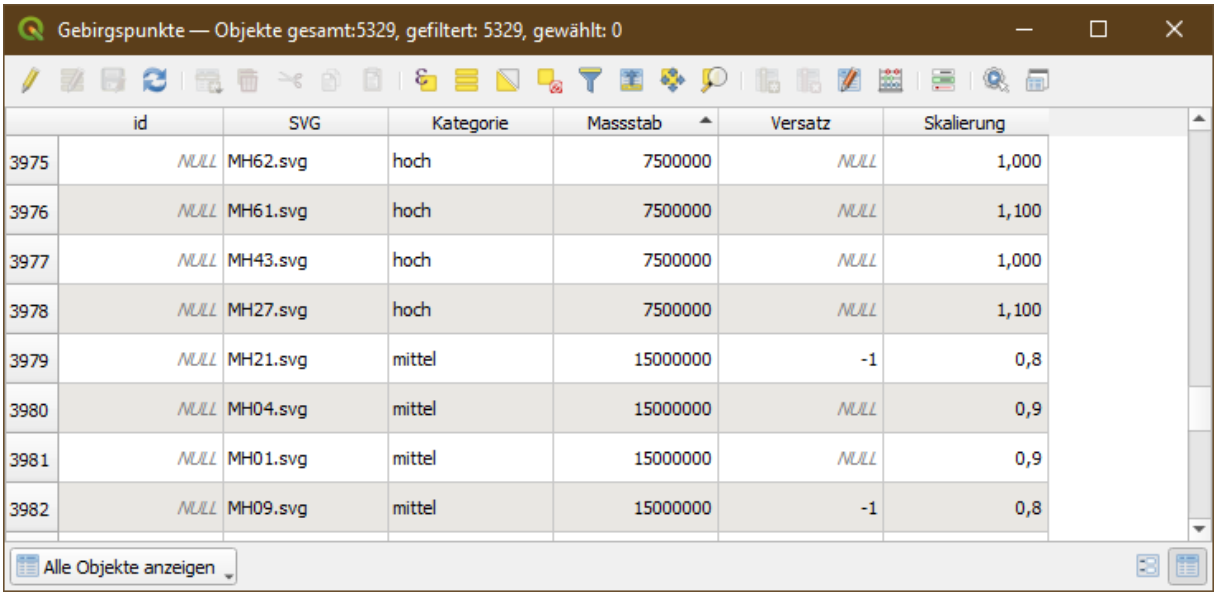
Entsprechend kann man mit allen sonstigen Parametern verfahren, die Abstände bezeichnen. Am Ende kamen bei mir folgende Ausdrücke heraus:

[5] Denn standardmäßig verdoppelt bzw. halbiert QGIS den Maßstab ja durch Scrollen, was allerdings unter *Einstellungen –> Optionen –> Kartenwerkzeuge –> Zoomen* auch geändert werden kann.

Zufallspunkte_hoch($scale/25000000+1200000/$scale, $scale/50000000+600000/$scale, $scale/70000000+2000000/$scale, 33, 42, $scale/70000000+500000/$scale, $scale)

Zufallspunkte_mittel($scale/33000000+1200000/$scale, $scale/65000000+1000000/$scale, $scale/60000000+3500000/$scale, 33, 42, $scale/70000000+500000/$scale, $scale)

VI. Darstellung der Punkte mit Hügelgrafiken

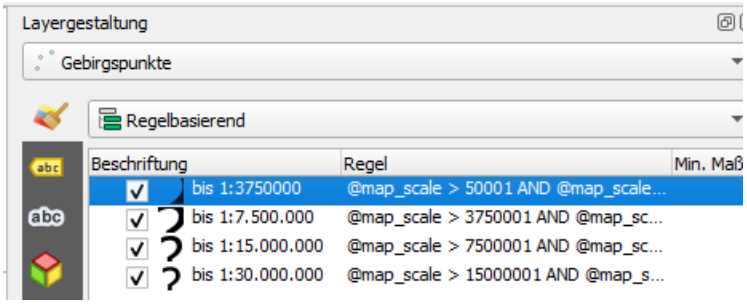


	id	SVG	Kategorie	Massstab	Versatz	Skalierung
3975	NULL	MH62.svg	hoch	7500000	NULL	1,000
3976	NULL	MH61.svg	hoch	7500000	NULL	1,100
3977	NULL	MH43.svg	hoch	7500000	NULL	1,000
3978	NULL	MH27.svg	hoch	7500000	NULL	1,100
3979	NULL	MH21.svg	mittel	15000000	-1	0,8
3980	NULL	MH04.svg	mittel	15000000	NULL	0,9
3981	NULL	MH01.svg	mittel	15000000	NULL	0,9
3982	NULL	MH09.svg	mittel	15000000	-1	0,8

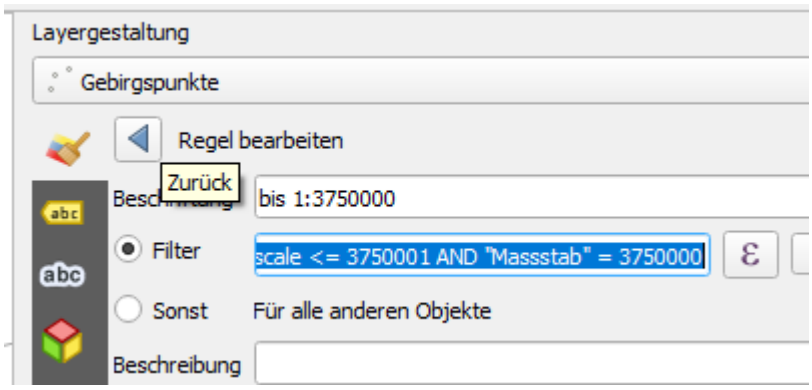
Alle generierten Punkte sind nun in die Attributtabelle der Ebene ‚Gebirgspunkte' eingetragen, wo sie sich in ihren jeweiligen Attributen unterscheiden: In der Spalte „SVG" ist hinterlegt, welche der 63 Maulwurfshügel-Dateien verwendet werden sollen. Aus der Spalte „Kategorie" ergibt sich, ob es sich um einen Punkt fürs Hoch- oder fürs Mittelgebirge handelt. Unter „Maßstab" steht, für welchen Maßstab der Punkt eingetragen wurde – nur für diesen soll er später auch sichtbar sein. „Versatz" weist für manche Punkte einen möglichen Versatz nach Ost oder West aus: So liegen z.B. die oben mit „-1" versehenen Punkte zu nahe am Westufer eines Flusses und sollten darum noch ein Stück weiter nach Westen verschoben werden. Aus der letzten Spalte schließlich ergibt sich ein Skalierungsfaktor, der für zwischen Flüssen ‚eingeklemmte' Punkte (s. Anhang) „0,5" beträgt, für zu verdrängende Punkte (s. Spalte ‚Versatz') „0,8" und für alle übrigen Punkte einen zufälligen Wert von 0,9 bis 1,1 (in Zehntelschritten. Bei Bedarf kann dies im Funktionscode geändert werden, s. Anhang.)

1. Maßstabsabhängige Sichtbarkeit der einzelnen Punkte

Sorgen wir also zunächst dafür, dass jeder Punkt nur für denjenigen Maßstab angezeigt wird, für den er erstellt wurde, oder, genauer gesagt, für einen Bereich von diesem Maßstab bis zum nächsten. Dafür bietet sich die *regelbasierende Symbolisierung* an:



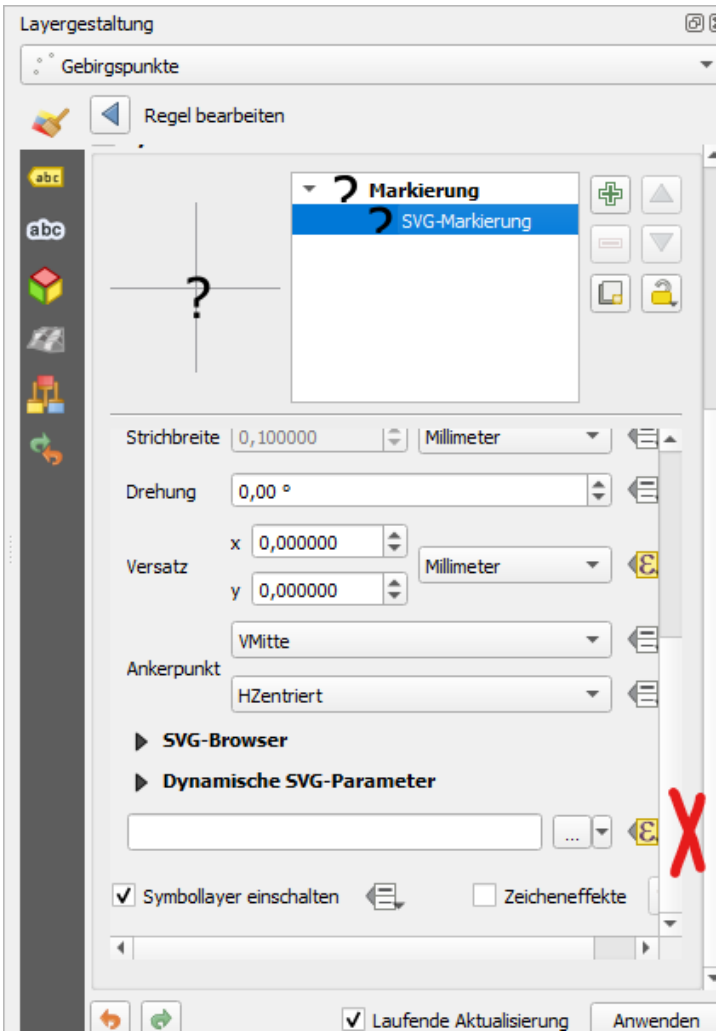
Im vorliegenden Beispiel wurden Punkte für die vier Maßstäbe 1:3750000, 1:7500000, 1:15000000 und 1:30000000 generiert. Dementsprechend definieren wir vier Anzeige-Regeln für die Bereiche von einem Maßstab bis zum nächsten, d.h. zunächst von einer willkürlich gegriffenen Untergrenze von 50001 bis zum Maßstabs-Kehrwert von 3750000, sodann von 3750000 bis 7500000 usw. Für jede dieser Regeln wird unter *Filter* ein Ausdruck eingetragen, der die Sichtbarkeit der Punkte regelt. Für die Regel „bis 1:3750000“ lautet er z.B.:




```
@map_scale > 50001 AND @map_scale <= 3750000 AND "Massstab" = 3750000
```

Das bedeutet: Ist der aktuelle Maßstabskehrwert größer als 50001 und zugleich kleiner od. gleich 3750000, und ist für den jeweiligen Punkt unter „Massstab“ „3750000“ eingetragen, so wird dieser Punkt angezeigt; wenn nicht, dann nicht.

Entsprechend wären auch die Regeln für die anderen Maßstabs-Bereiche zu formulieren.



2. Zuweisung der SVG-Dateien

Wenn wir doppelt auf die einzelne Regel klicken und dann noch einmal auf *SVG-Markierung*, so findet sich ganz unten ein Feld, wo der Pfad zu einer SVG-Datei eingetragen werden kann. Da wir jedoch nicht ein und dieselbe Hügelgrafik für sämtliche Punkte verwenden, sondern jedem Punkt die Grafik zuweisen wollen, die für ihn in der Attributtabelle hinterlegt ist, müssen wir stattdessen rechts von diesem Feld den ε-Knopf drücken, um einen Ausdruck wie den folgenden eintragen zu können:

```
'd:/Kartografie/Signaturen/Maulwurfshügel/' || "SVG"
```

Er gibt zunächst den Pfad zur SVG-Datei wieder, ergänzt um dasjenige, was in der Spalte „SVG“ steht, also den Dateinamen, – mithin insgesamt einen vollständigen Pfad wie zum Beispiel „D:\Kartografie\Signaturen\Maulwurfshügel\MH31.svg“.

3. Skalierung der Hügelgrafiken abhängig vom Maßstab – und vom Breitengrad

Nach Doppelklick auf eine der Regeln und einfachem Klick auf *SVG- Markierung* finden wir unten auch die Felder, um die Größe der Hügelgrafiken festzulegen. Hier sollte zunächst eine passende Hügelgröße für den nominalen Maßstab der jeweiligen Regel eingestellt werden. (Wir sollten also darauf achten, dass die Karte sich aktuell in diesem Maßstab gefindet und nicht irgendwo zwischen diesem und dem nächsten.) Die angegebene Größe bezieht sich dabei auf den unsichtbaren Kreis, der im Hintergrund einer jeden Hügelgrafik liegt.

Wenn wir ein ansprechendes Kartenbild gefunden haben, sollten wir dafür sorgen, dass dieses auch dann erhalten bleibt, wenn die Karte in den Bereich zwischen diesem und dem nächsten Maßstab hinein vergrößert wird. Die Hügel sollten dabei nämlich mit vergrößert werden, um ihre relative Größe im Verhältnis zum übrigen Kartenbild zu behalten. Das erreichen wir, indem den ε-Knopf drücken und mit *Bearbeiten* den Ausdrucksеditor öffnen, wo folgender Ausdruck eingetragen werden kann:

```
CASE
  WHEN "Kategorie" = 'hoch' THEN 8 * "Skalierung" * 7500000/$scale * (-0.021 * $y +2.1)
  WHEN "Kategorie" = 'mittel' THEN 7.5* "Skalierung" * 7500000/$scale * (-0.021 * $y +2.1)
  ELSE 1
END
```

Seine ersten Zeilen bedeuten: Im Fall wenn die Kategorie „hoch“ ist, dann hat der Punkt eine Größe von 8 mm, die jedoch skaliert wird um den Faktor aus der Spalte „Skalierung“ (damit das Hügelbild abwechslungsreicher wird), anschließend um einen maßstabsabhängigen Faktor, der im Nominal-Maßstab genau 1 beträgt, sich jedoch beim Vergrößern des Maßstabs mit vergrößert, und schließlich um den hinteren, eingeklammerten Faktor, welcher den y-Wert auf der Karte berücksichtigt, also den Breitengrad. Denn da die Koordinatenbezugssysteme der vorliegenden Ebenen Grad als Karteneinheit haben, sind auch die Hügel-Abstände in Grad angegeben – einer Einheit also, die umso länger ist, je näher man dem Äquator kommt. Das Kartenbild wird darum etwas schöner, wenn die Hügel nach Süden hin im gleichen Maße größer werden wie die Grade. Bei dem hier verwendeten Koordinatenbezugssystem EPSG:3034 mit seiner konischen Lambert-Projektion divergieren die Meridiane in südlicher Richtung linear, was es einfach macht, eine Formel für eine ebenso lineare Vergrößerung der Hügel zu finden (nämlich „-0.021 * $y +2.1“. Das absolute Glied wurde hierbei so gewählt, dass die Hügel etwa beim 50. Breitengrad ihre eigentliche Größe haben und von dort aus nordwärts kleiner und südwärts größer werden.)

Für die Mittelgebirgshügel verhält es sich in der dritten Zeile entsprechend; sie erhalten eine grundsätzliche Größe von 7,5 mm. Der Auffangwert von 1 in der vorletzten Zeile wird nur relevant, falls unter „Kategorie“ nichts eingetragen ist (also eigentlich nie).

4. Abrücken der Hügel von Flüssen, Küstenlinien etc.

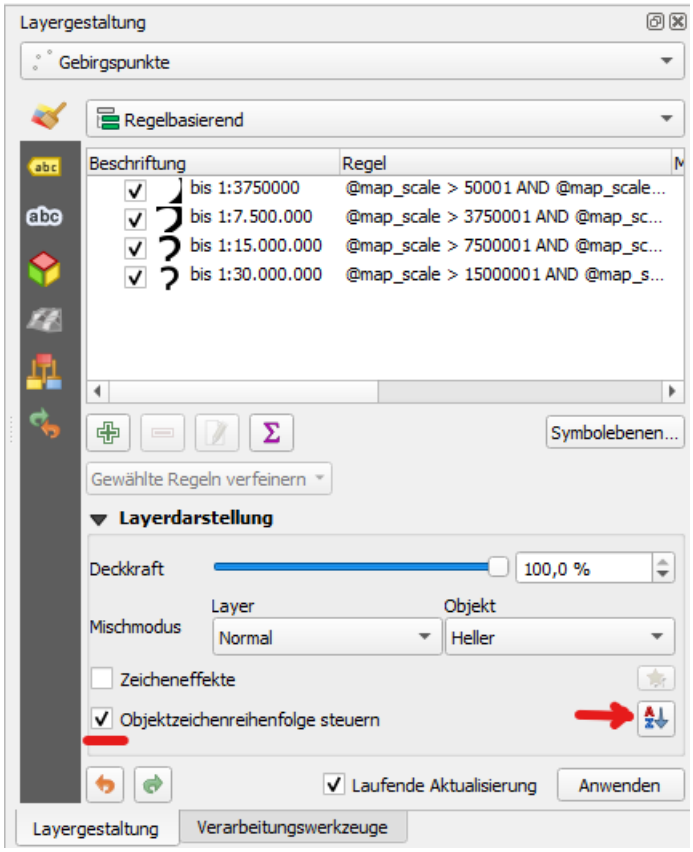
Ebenfalls an der schon mehrfach erwähnten Stelle findet sich die Rubrik *Versatz,* wo anstatt eines festen Versatzes für die Hügelgrafiken auch einen solcher eingestellt werden kann, der sich nach den Einträgen in der Attributspalte „Versatz“ richtet. Dafür drücken wir wieder den ε-Knopf, gehen auf *Bearbeiten* und geben im Ausdrucksеditor z.B. Folgendes ein:

```
to_string("Versatz"*0.7) || ',' || 0
```

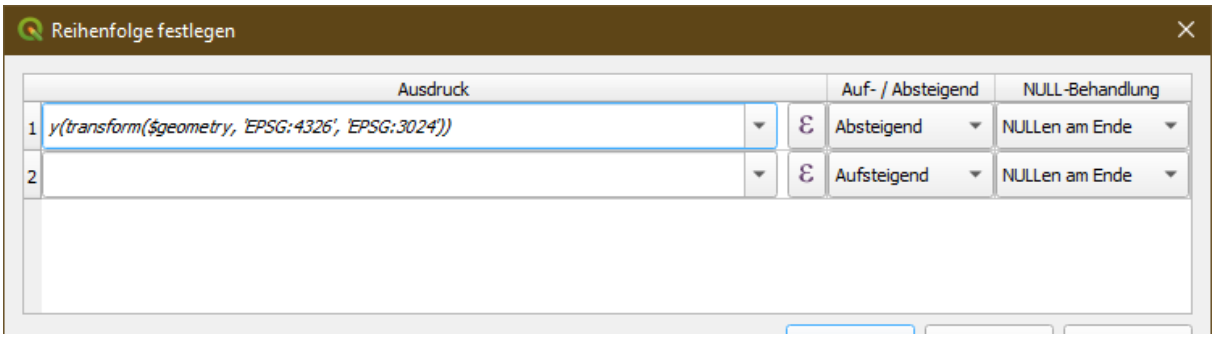
Dieser Ausdruck gibt einen Versatz im Format „x,y“ wieder. Als x-Wert wird dabei der Eintrag aus der Spalte „Versatz“ genommen und mit 0,7 multipliziert (einem Faktor, der frei wählbar ist). Das Ergebnis wird in einen ‚String‘ umgewandelt, um für QGIS verdaulich zu sein. Fort fährt die Verkettung mit einem Komma als Trennzeichen und einer Null als y-Wert.

5. Justieren der Zeichenreihenfolge fürs aktuelle Koordinatenreferenzsystem

Damit die Hügeldarstellung einen perspektivischen Eindruck erweckt, sind die beiden Funktionen schon so formuliert, dass die Hügel nach ihrem y-Wert sortiert und die südlicheren Hügel erst nach den nördlicheren gezeichnet werden, damit die vorderen Hügel die hinteren verdecken und nicht umgekehrt (s. Anhang). Damit sieht das Kartenbild schon halbwegs ordentlich aus. Als y-Wert dient hier aber der geografische Nord-/Süd-Wert, und dies führt dort zu unzureichenden Ergebnissen, wo Norden in der Karte nicht (oder nicht ganz) oben ist: nämlich in den östlichen und westlichen Randbereichen, wo die Meridiane schräg zum Nordpol hin verlaufen, und erst recht in Karten, die gar nicht genordet sind, sondern z.B. geostet (wie es in alten Zeiten nicht unüblich war).



Um ein wirklich ordentliches Kartenbild zu erhalten, müssen wir also erreichen, dass die Hügel in der Karte wirklich von oben nach unten gezeichnet werden. Das geht, indem wir in der *Layergestaltung* ganz unten das Häkchen bei *Objektzeichenreihenfolge steuern* setzen und rechts davon den Knopf drücken. Nun öffnet sich folgendes Fenster,



in welches wir folgenden Ausdruck eintragen können:

y(transform($geometry, 'EPSG:4326', 'EPSG:3024'))

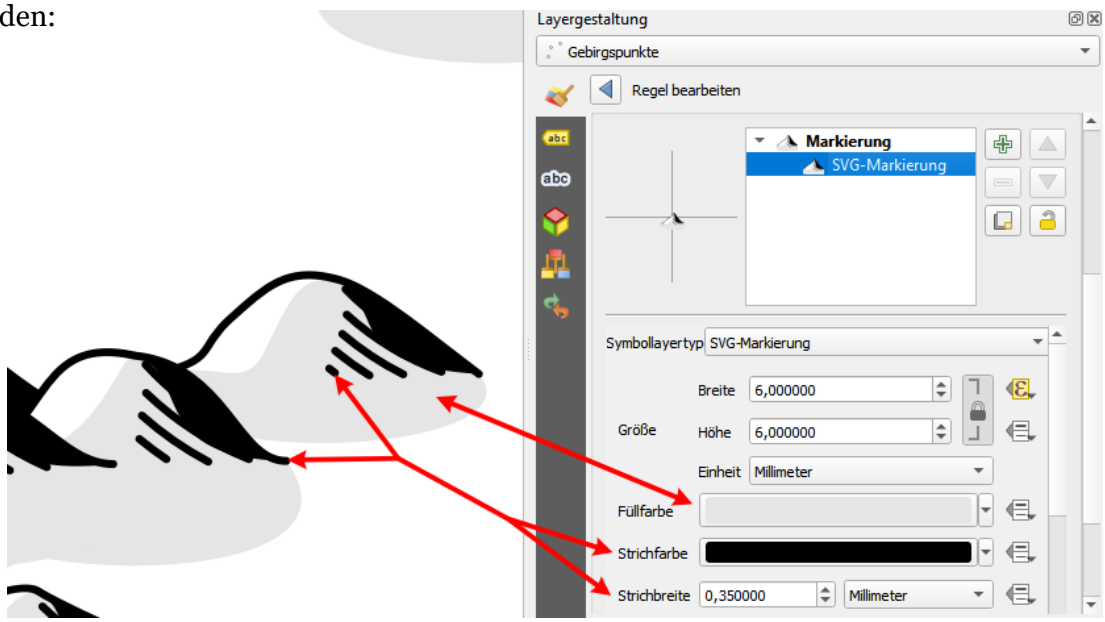
Er bewirkt, dass die Punkte in der Reihenfolge ihres y-Werts gezeichnet werden, jedoch erst, nachdem die Koordinaten der Ebene (EPSG:4326) in die Koordinaten am Bildschirm (EPSG:3024) transformiert wurden.

6. Einstellen von Farbe und Strichstärke der Hügel

Zum Schluss geht es noch darum, Farben und Strichstärken der Hügelgrafiken einzustellen. Diese sind in den SVG-Dateien nämlich nicht fix hinterlegt, sondern als Variablen (Parameter) offen gelassen, deren Wert durch QGIS bestimmt werden kann (sog. ‚parametrisierte' SVG-Dateien). Das bedeutet, dass wir in QGIS das Aussehen der Hügelgrafiken an die Bedürfnisse unseres Kartenbildes anpassen können.

Um die nötigen Felder zu aktivieren, die in QGIS bisher gegraut sind, müssen wir allerdings doch noch so tun, als wollten wir ein und dieselbe Grafik für alle Hügel verwenden und an der auf S. 10 angekreuzten Stelle den Pfad zu einer konkreten Hügeldatei angeben:

(Welche wir wählen, ist egal, da es sich nicht auswirkt, sondern nach wie vor datendefiniert übersteuert wird.) Nun können Farben und Strichstärken mit folgenden Feldern eingestellt werden:



Klickt man auf das Farbfeld bei *Füllfarbe,* eröffnen sich genauere Einstellungsmöglichkeiten, u.a. hinsichtlich der Deckkraft. Hier ist zu beachten, dass QGIS die Deckkraft der Hügelfarbe einzustellen glaubt, während es sich in Wahrheit auf den schwarzen Schatten auswirkt! Diese Zweckentfremdung war notwendig, weil QGIS' Nutzung parametrisierter SVG-Dateien derzeit noch etwas defizitär ist.