GIS - Task #8384

Applying Topology Rules to Basins using QGIS 3.6.*

28/05/2019 16:16 - Debojyoti Mallick

Status:	Feedback	**Start date:**	28/05/2019
Priority:	Normal	**Due date:**	
Assignee:	Debojyoti Mallick	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

Applying Topology Rules to refine borders of Basins using QGIS.

Used Check Geometry and Topology Checker to try and correct ends of basins.

Some Basins looked good and followed the correct rule for overlap while a few of them did not.

Checking through all the rules to eventually look at the possibility of removing the faulty overlaps automatically. Else will have to resort to manual editing for them.

History

#1 - 07/06/2019 12:29 - Debojyoti Mallick

- File AfterCheckGeometry.png added

- File AfterDissolve2.png added

- File AttributeforDissolve.png added

- File BeforeCheckGeometry.png added

- File BeforeDissolve2.png added

- File Dissolve Problem1.png added

- File JoinwithDrain.png added

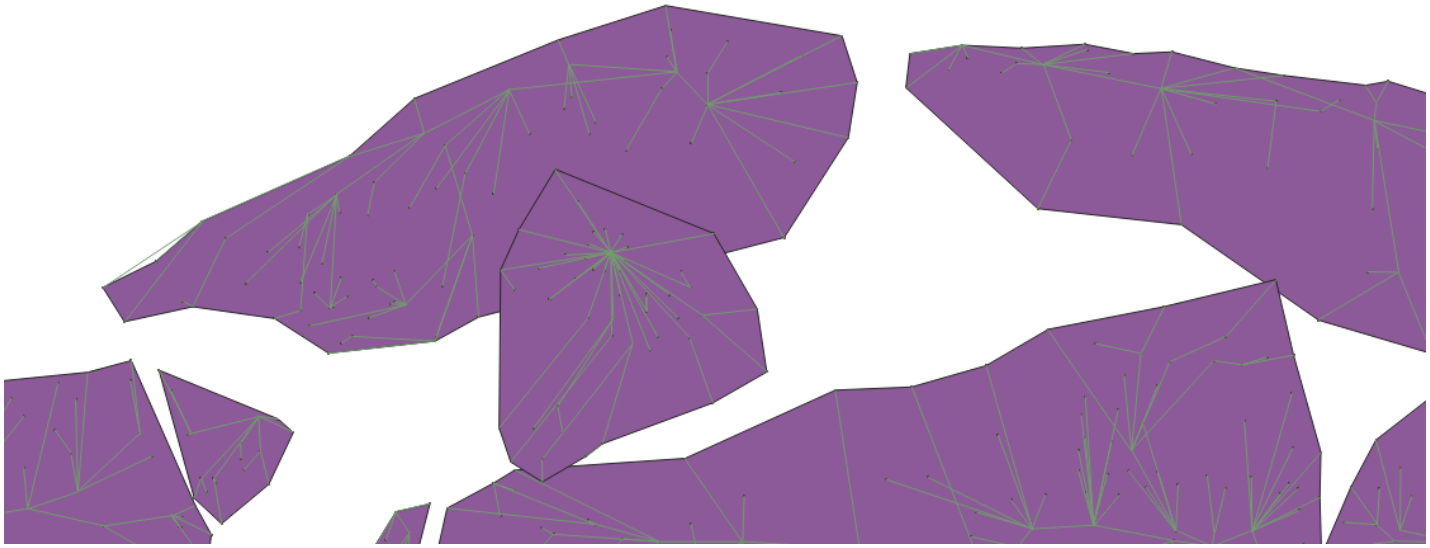
- File Dissolve.png added

Debojyoti Mallick wrote:

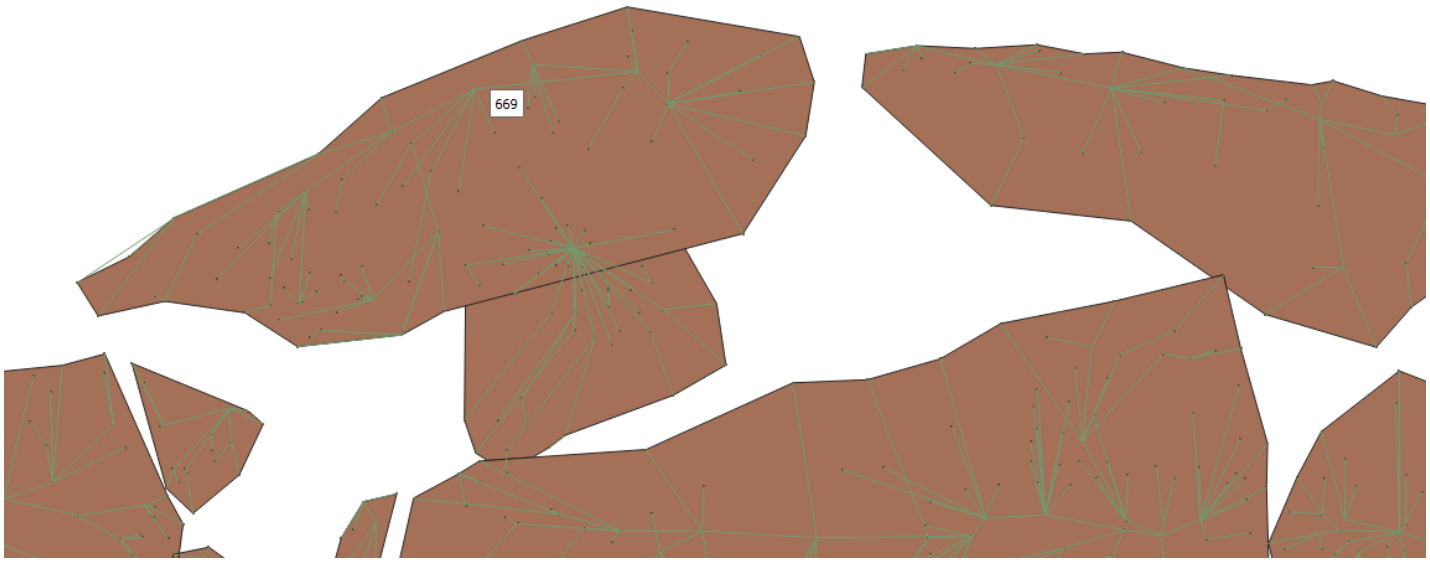
> Applying Topology Rules to refine borders of Basins using QGIS.
>
> Used Check Geometry and Topology Checker to try and correct ends of basins.
>
> Some Basins looked good and followed the correct rule for overlap while a few of them did not.
>
> Checking through all the rules to eventually look at the possibility of removing the faulty overlaps automatically. Else will have to resort to manual editing for them.

Check Geometry and Topology Checker cannot be used to correct overlaps as both of them define an overlap only considering how the boundaries of polygons are shared with one another.
This defeats the purpose as the overlaps should be eliminated by the logic of the catchment points inside the basin and the direction of the drain to the lowest catchment area.

Only using simple Topology checks results in such anomalies.



Before using Check Geometry



As you can see in the image above the overlaps are removed but they do not respect the direction of the drains nor do they respect the location of the catchment points.

Second alternative which was used was VClean using GRASS toolbox.

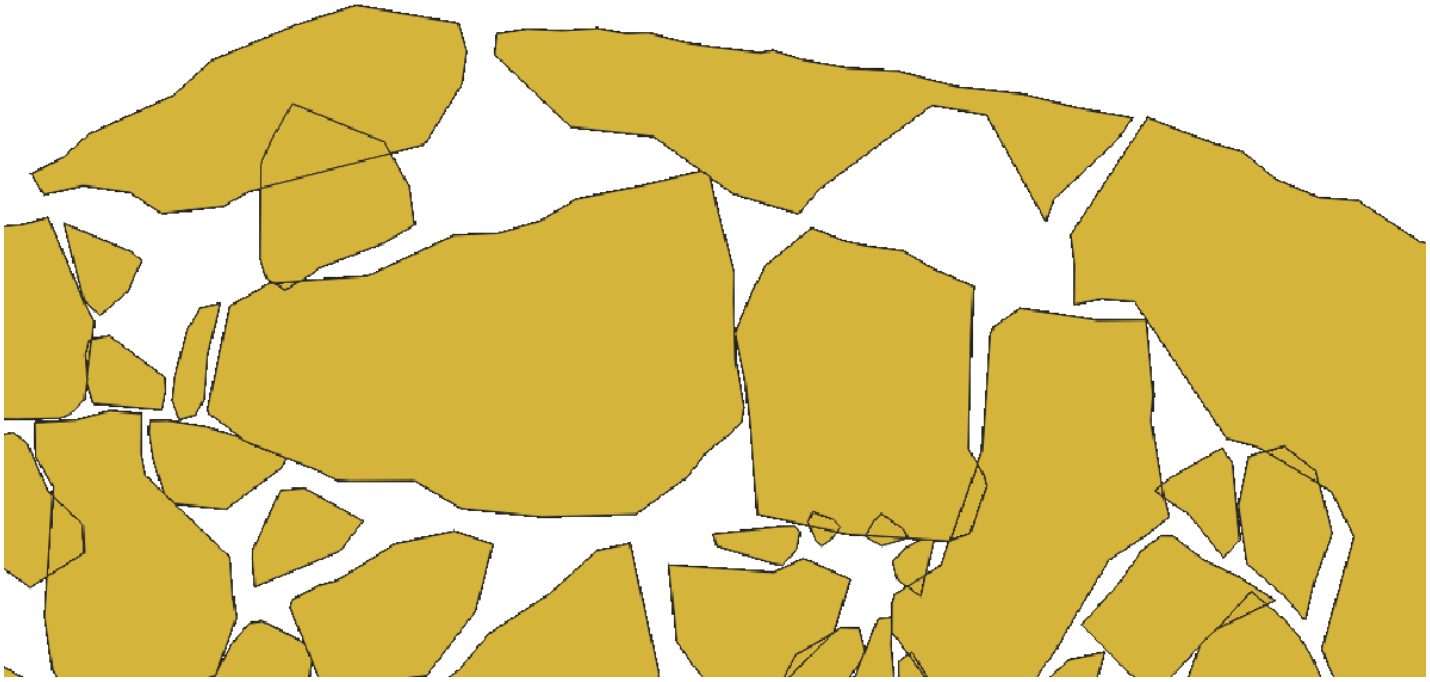
This uses the Topology Checker results to eventually clean the overlaps. This does give much better results but still creates some similar anomalies as discussed before. In addition to this it also automatically eliminates Silver Polygons which is not ideal as it would eventually eliminate the sub-basins and very small catchment areas.

To resolve geometric errors inclusive of Drains and Catchment points we did not use simple Topology rules as QGIS does not give the option of relations between points, lines and polygons.

Hence we use basic spatial concepts like Spatial Joins, Merge and Dissolve.

We first run a Spatial Join (Join by Location) between the Basins (Polygons) and the Drains (Lines)
A single attribute table with both results gives us more control.

The resulting layer is as follows.

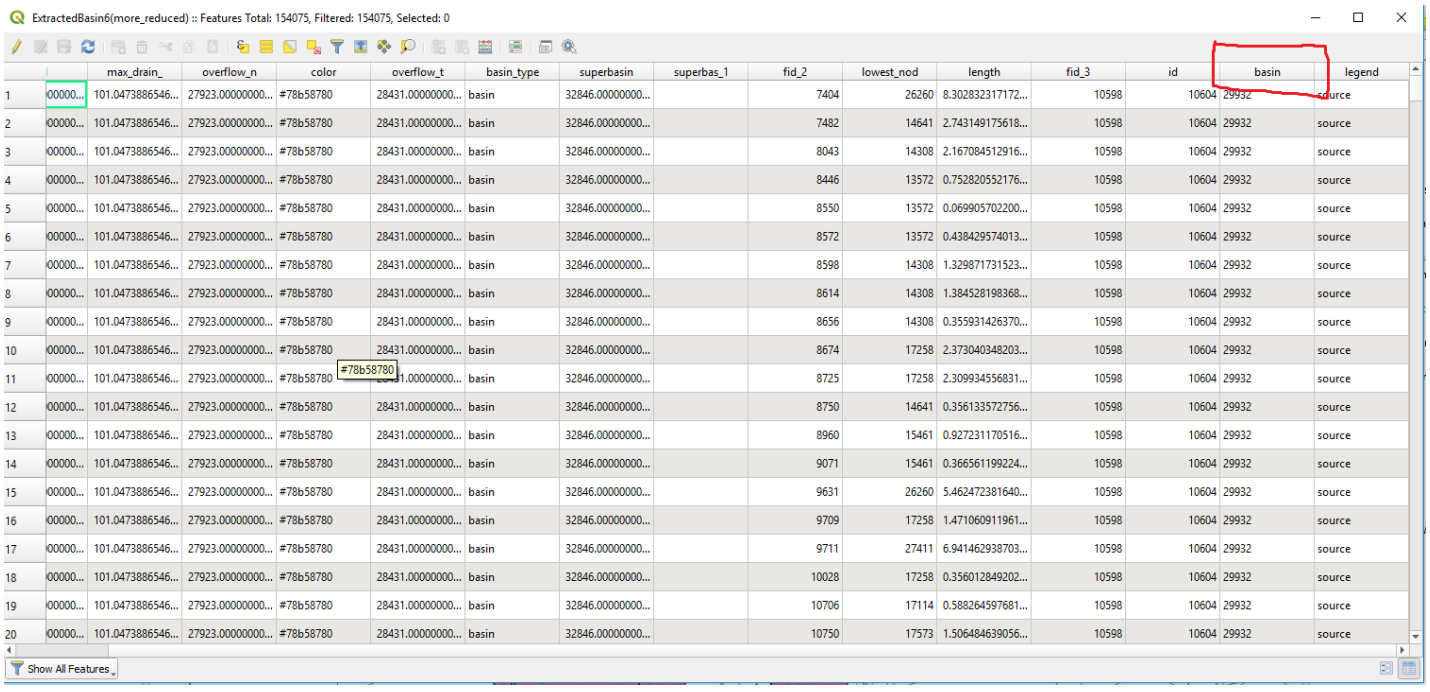


As the above figure shows, we can now see all the defined shapes of the basins and this will not show any overlaps.
This is now used as a base layer and we use the drain numbers to run our dissolve.

The resulting layer is not correct as consecutive drain numbers do not necessarily lead to or belong to the same basin.

Hence we have to include the catchment points and use those as a reference to define the boundaries for the basins.
On checking the attribute table the Basin column in the catchment points is a very reliable attribute to work with.

Hence ran a Spatial Join between the Basins, Drains and the Catchment Points.


ExtractedBasin6(more_reduced) :: Features Total: 154075, Filtered: 154075, Selected: 0

	max_drain_	overflow_n	color	overflow_t	basin_type	superbasin	superbas_1	fid_2	lowest_nod	length	fid_3	id	basin	legend
1	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		7404	26260	8.302832317172...	10598	10604	29932	source
2	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		7482	14641	2.743149175618...	10598	10604	29932	source
3	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8043	14308	2.167084512916...	10598	10604	29932	source
4	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8446	13572	0.752820552176...	10598	10604	29932	source
5	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8550	13572	0.069905702200...	10598	10604	29932	source
6	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8572	13572	0.438429574013...	10598	10604	29932	source
7	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8598	14308	1.329871731523...	10598	10604	29932	source
8	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8614	14308	1.384528198368...	10598	10604	29932	source
9	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8656	14308	0.355931426370...	10598	10604	29932	source
10	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8674	17258	2.373040348203...	10598	10604	29932	source
11	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8725	17258	2.309934556831...	10598	10604	29932	source
12	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8750	14641	0.356133572756...	10598	10604	29932	source
13	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8960	15461	0.927231170516...	10598	10604	29932	source
14	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9071	15461	0.366561199224...	10598	10604	29932	source
15	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9631	26260	5.462472381640...	10598	10604	29932	source
16	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9709	17258	1.471060911961...	10598	10604	29932	source
17	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9711	27411	6.941462938703...	10598	10604	29932	source
18	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10028	17258	0.356012849202...	10598	10604	29932	source
19	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10706	17114	0.588264597681...	10598	10604	29932	source
20	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10750	17573	1.506484639056...	10598	10604	29932	source

Show All Features

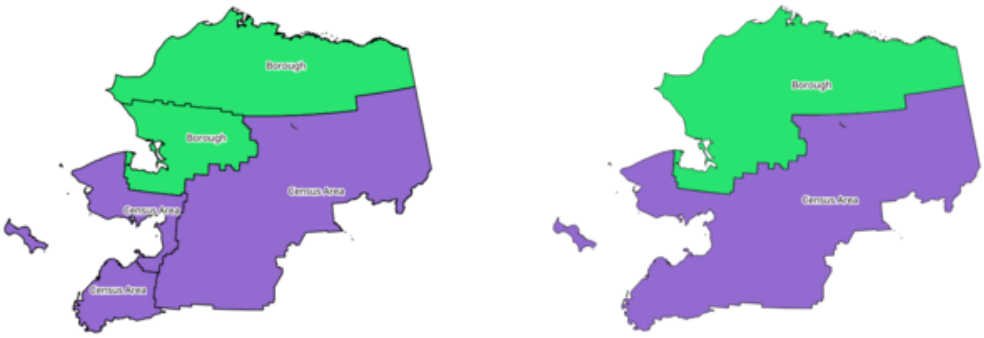
We now run GDAL Dissolve using the Catchment Points as a reference.

The results are good but we do have a few anomalies as the tool cannot decide the borders of the certain basins.

!Dissolve Problem1.png!

As per the image above, you will see that the polygons have been merged because the dissolve tool was not able to decide where the point belongs.
But this is the only anomaly with regards to a single point.

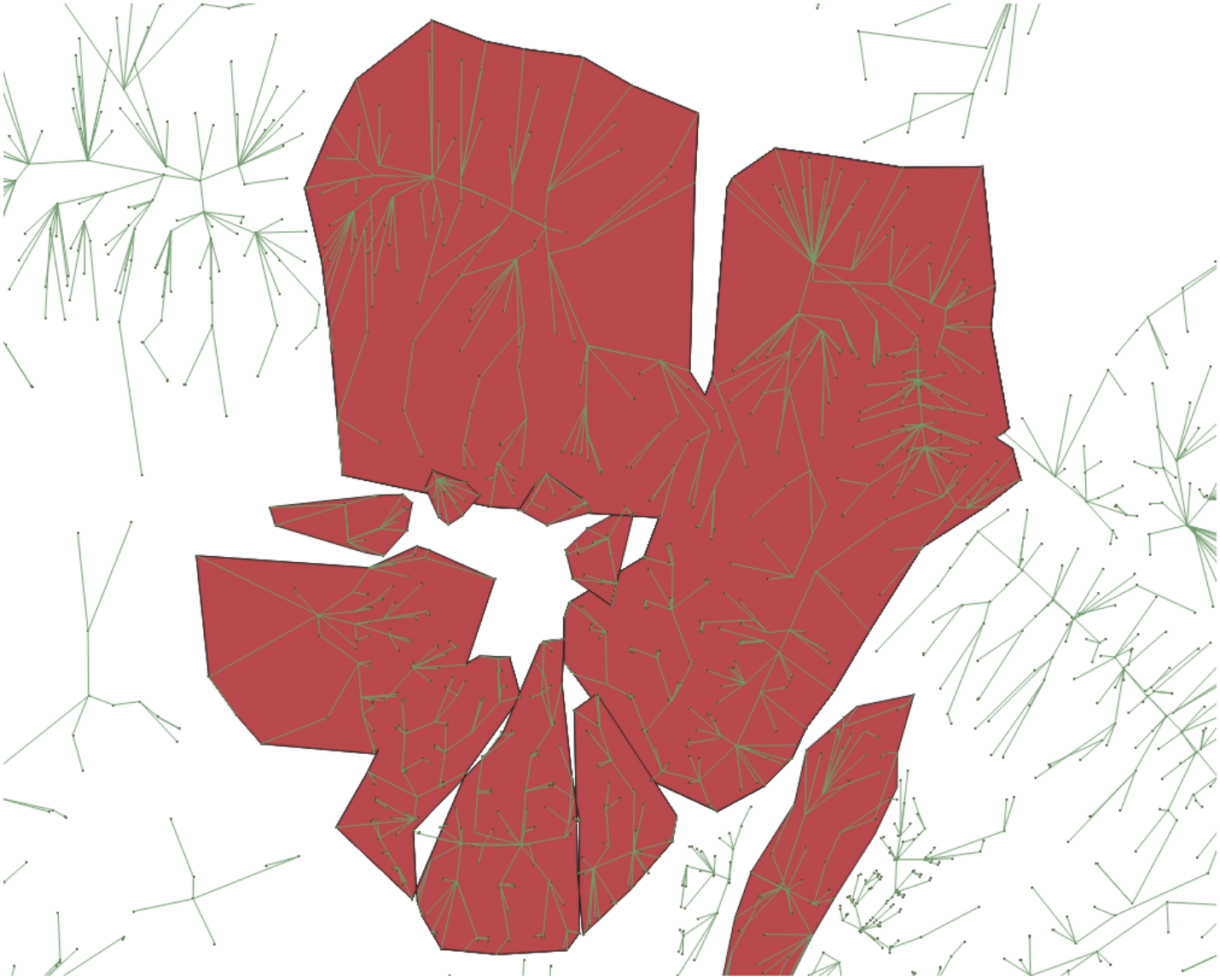
Keep in mind that we have considered the true definition and use of Dissolve illustrated below.



Dissolve the polygon layer on a common attribute

But the difference here is the fact that we are using the Catchment Points as the defining variable hence it should not merge the polygons rather define the borders of the basins (by eradicating unnecessary overlaps).

A similar issue of Merging Polygons occurs when there are multiple catchment points between two polygons.
The GDAL Dissolve tool is unable to decide the correct outline of the basin in such areas.



In the image above you can see another case of merged polygons.

The rest of the layer does not show this problem as 80% of the overlaps are eradicated correctly.

Hence we use Dissolve in Python/GeoPandas rather than use the GDAL tool in QGIS to check if the same issue is replicated.

There seems to be an issue with Jupyter with regards to reading files

gdf = gpd.read_file('SandBox/Debo/data/ExtractedBasin6more_reduced.shp')

Unfortunately files cannot be read. I will run this once it has been resolved.

Hence running it on VSCode to check the results. Takes time because of the hardware incompetence of my machine. Will be running this on a desktop to check the results.

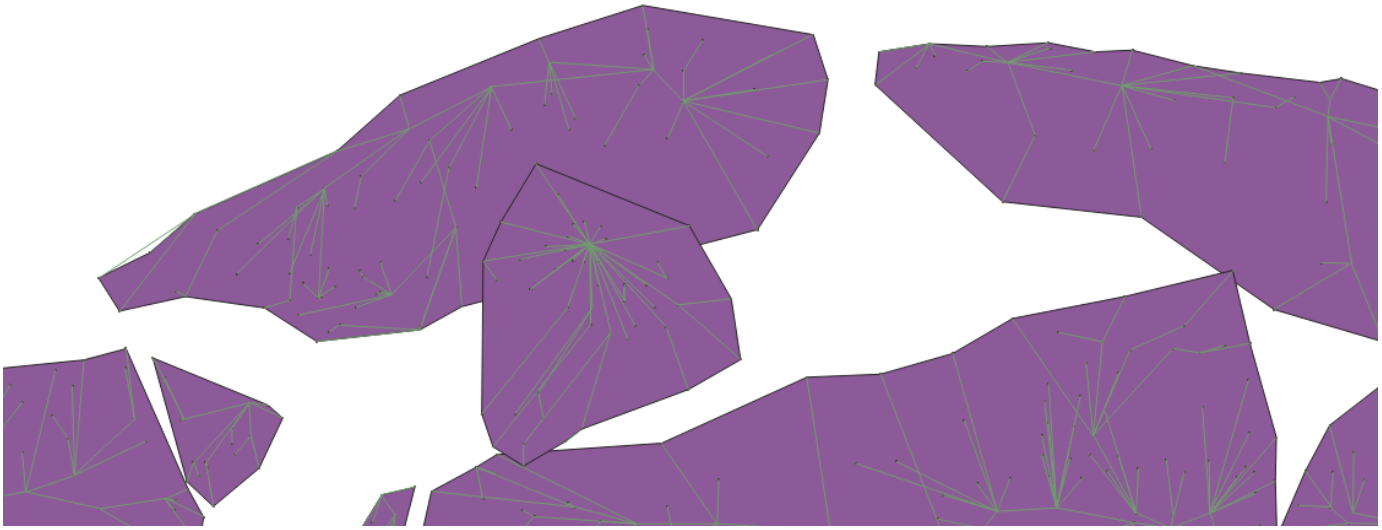
#2 - 07/06/2019 14:55 - Debojyoti Mallick

Debojyoti Mallick wrote:

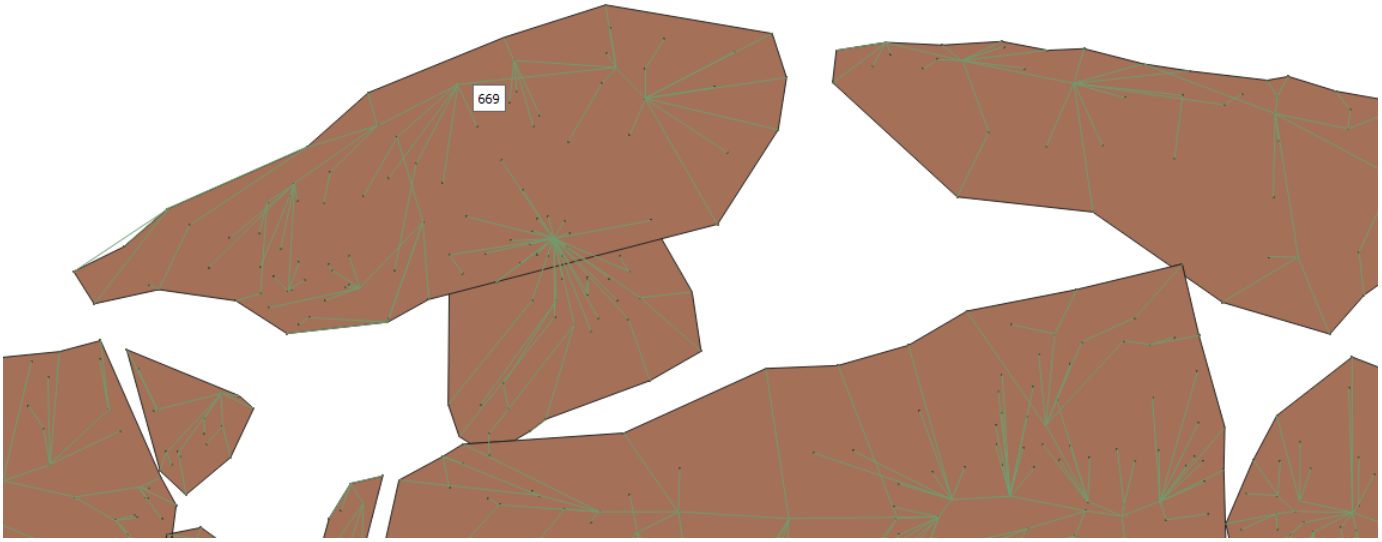
> Debojyoti Mallick wrote:
>
> > Applying Topology Rules to refine borders of Basins using QGIS.
> >
> > Used Check Geometry and Topology Checker to try and correct ends of basins.
> >
> > Some Basins looked good and followed the correct rule for overlap while a few of them did not.
> >
> > Checking through all the rules to eventually look at the possibility of removing the faulty overlaps automatically. Else will have to resort to manual editing for them.

Check Geometry and Topology Checker cannot be used to correct overlaps as both of them define an overlap only considering how the boundaries of polygons are shared with one another.
This defeats the purpose as the overlaps should be eliminated by the logic of the catchment points inside the basin and the direction of the drain to the lowest catchment area.

Only using simple Topology checks results in such anomalies.



Before using Check Geometry



As you can see in the image above the overlaps are removed but they do not respect the direction of the drains nor do they respect the location of the catchment points.

Second alternative which was used was VClean using GRASS toolbox.

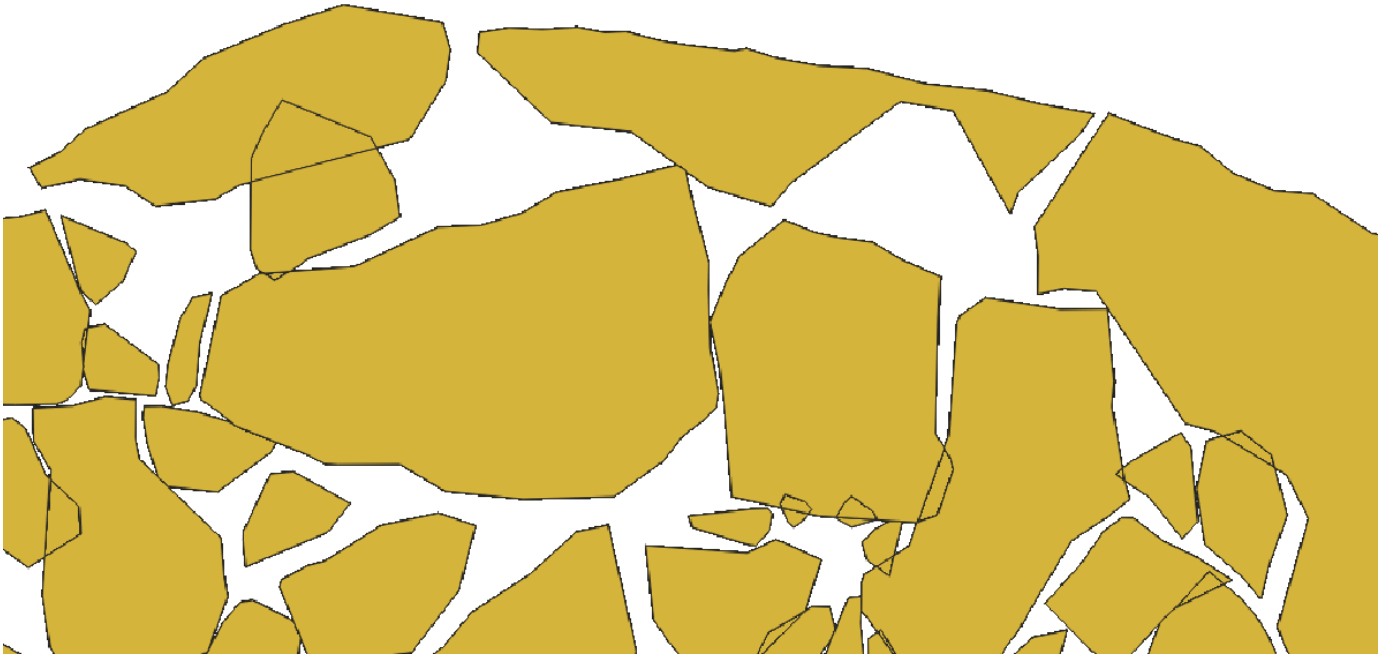
This uses the Topology Checker results to eventually clean the overlaps. This does give much better results but still creates some similar anomalies as discussed before. In addition to this it also automatically eliminates Silver Polygons which is not ideal as it would eventually eliminate the sub-basins and very small catchment areas.

To resolve geometric errors inclusive of Drains and Catchment points we did not use simple Topology rules as QGIS does not give the option of relations between points, lines and polygons.

Hence we use basic spatial concepts like Spatial Joins, Merge and Dissolve.

We first run a Spatial Join (Join by Location) between the Basins (Polygons) and the Drains (Lines)
A single attribute table with both results gives us more control.

The resulting layer is as follows.

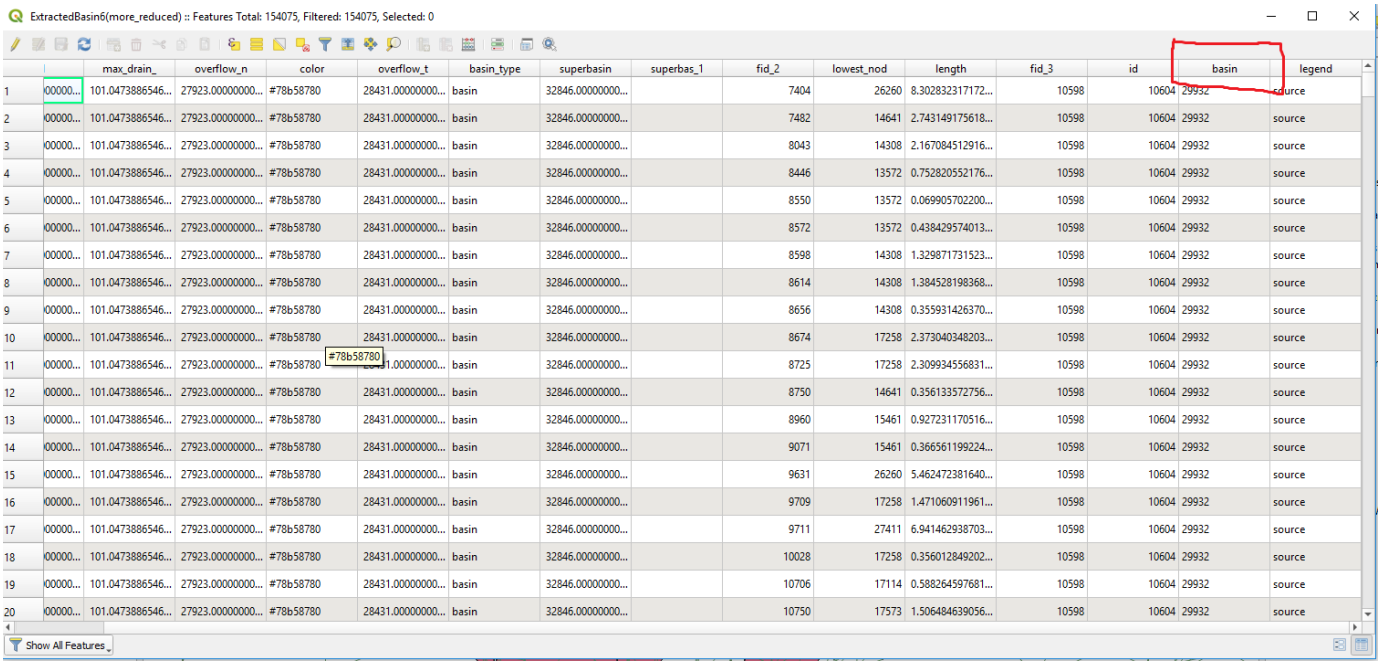


As the above figure shows, we can now see all the defined shapes of the basins and this will not show any overlaps.
This is now used as a base layer and we use the drain numbers to run our dissolve.

The resulting layer is not correct as consecutive drain numbers do not necessarily lead to or belong to the same basin.

Hence we have to include the catchment points and use those as a reference to define the boundaries for the basins.
On checking the attribute table the Basin column in the catchment points is a very reliable attribute to work with.

Hence ran a Spatial Join between the Basins, Drains and the Catchment Points.



	max_drain_	overflow_n	color	overflow_t	basin_type	superbasin	superbas_1	fid_2	lowest_nod	length	fid_3	id	basin	legend
1	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		7404	26260	8.302832317172...	10598	10604	29932	source
2	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		7482	14641	2.743149175618...	10598	10604	29932	source
3	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8043	14308	2.167084512916...	10598	10604	29932	source
4	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8446	13572	0.752820552176...	10598	10604	29932	source
5	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8550	13572	0.069905702200...	10598	10604	29932	source
6	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8572	13572	0.438429574013...	10598	10604	29932	source
7	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8598	14308	1.329871731523...	10598	10604	29932	source
8	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8614	14308	1.384528198368...	10598	10604	29932	source
9	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8656	14308	0.355931426370...	10598	10604	29932	source
10	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8674	17258	2.373040348203...	10598	10604	29932	source
11	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8725	17258	2.309934556831...	10598	10604	29932	source
12	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8750	14641	0.356133572756...	10598	10604	29932	source
13	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8960	15461	0.927231170516...	10598	10604	29932	source
14	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9071	15461	0.366561199224...	10598	10604	29932	source
15	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9631	26260	5.462472381640...	10598	10604	29932	source
16	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9709	17258	1.471060911961...	10598	10604	29932	source
17	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9711	27411	6.941462938703...	10598	10604	29932	source
18	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10028	17258	0.356012849202...	10598	10604	29932	source
19	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10706	17114	0.588264597681...	10598	10604	29932	source
20	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10750	17573	1.506484639056...	10598	10604	29932	source

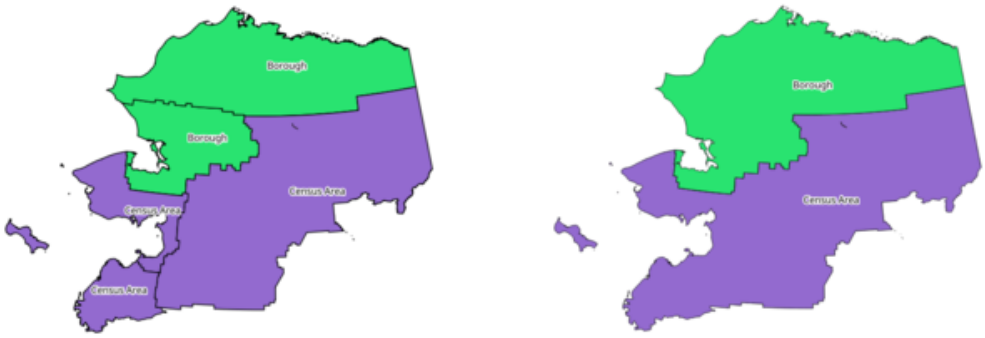
We now run GDAL Dissolve using the Catchment Points as a reference.

The results are good but we do have a few anomalies as the tool cannot decide the borders of the certain basins.

!Dissolve Problem1.png!

As per the image above, you will see that the polygons have been merged because the dissolve tool was not able to decide where the point belongs. But this is the only anomaly with regards to a single point.

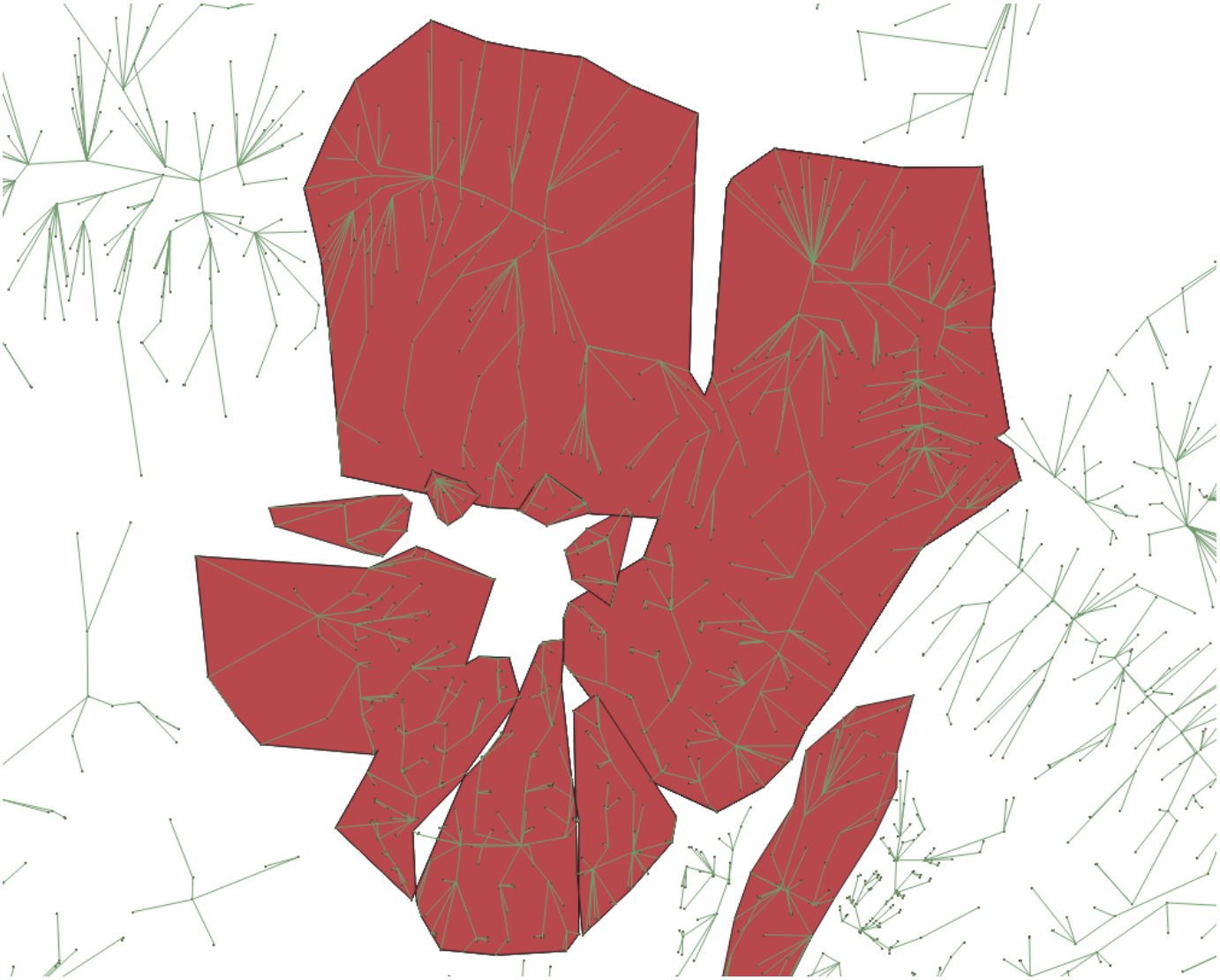
Keep in mind that we have considered the true definition and use of Dissolve illustrated below.



Dissolve the polygon layer on a common attribute

But the difference here is the fact that we are using the Catchment Points as the defining variable hence it should not merge the polygons rather define the borders of the basins (by eradicating unnecessary overlaps).

A similar issue of Merging Polygons occurs when there are multiple catchment points between two polygons.
The GDAL Dissolve tool is unable to decide the correct outline of the basin in such areas.



In the image above you can see another case of merged polygons.

The rest of the layer does not show this problem as 80% of the overlaps are eradicated correctly.

Hence we use Dissolve in Python/GeoPandas rather than use the GDAL tool in QGIS to check if the same issue is replicated.

There seems to be an issue with Jupyter with regards to reading files

gdf = gpd.read_file('SandBox/Debo/data/ExtractedBasin6more_reduced.shp')

Unfortunately files cannot be read. I will run this once it has been resolved.

Hence running it on VSCode to check the results. Takes time because of the hardware incompetence of my machine. Will be running this on a desktop to check the results.

Ran the dissolve using GeoPandas on Jupyter using a geopackage. Same results worried.png

This was the code used:

```
import geopandas
gdf = geopandas.read_file('data/Extractedbasin6_geopack.gpkg').to_crs({'init' :'epsg:4326'})

import numpy as np

gdf.plot(color='white', edgecolor='black')

import matplotlib.pyplot as plt
import matplotlib.lines as mlines
from matplotlib.colors import ListedColormap
```

gdf.geom_type.head()

Basin_bound = gdf['basin', 'geometry'

Basin_dissolve = Basin_bound.dissolve(by='basin')

Basin_dissolve

Basin_dissolve.plot(color='white', edgecolor='black')

#Gives the same result

This concludes that there is no issue with the GDAL Dissolve and the logic used to get the output is flawed.

Will check for alternatives.

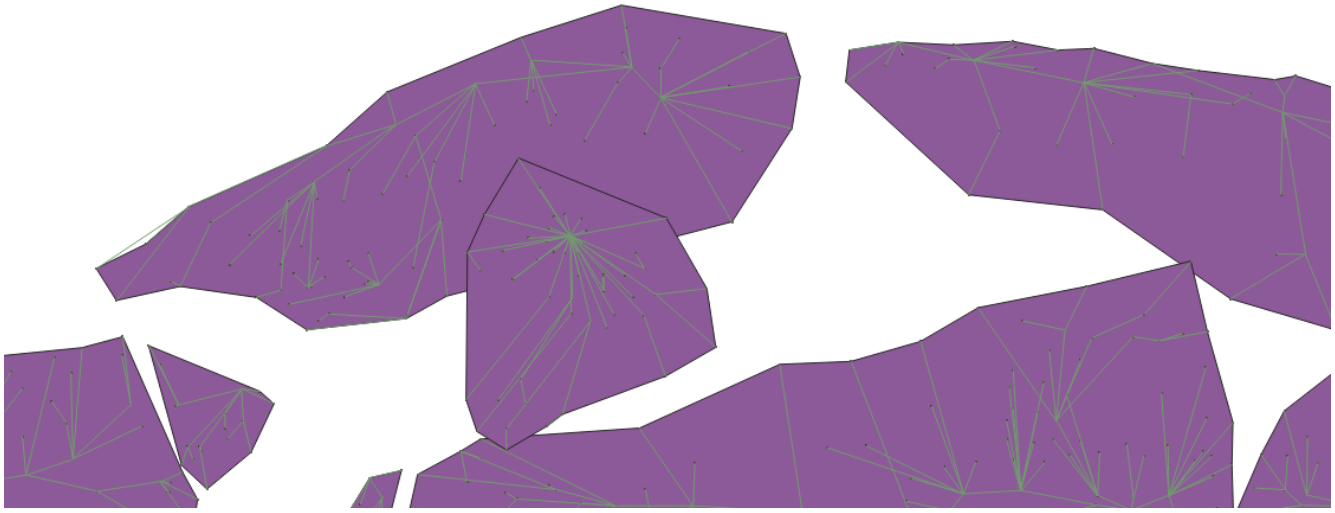
#3 - 07/06/2019 15:29 - Debojyoti Mallick

- File After.png added

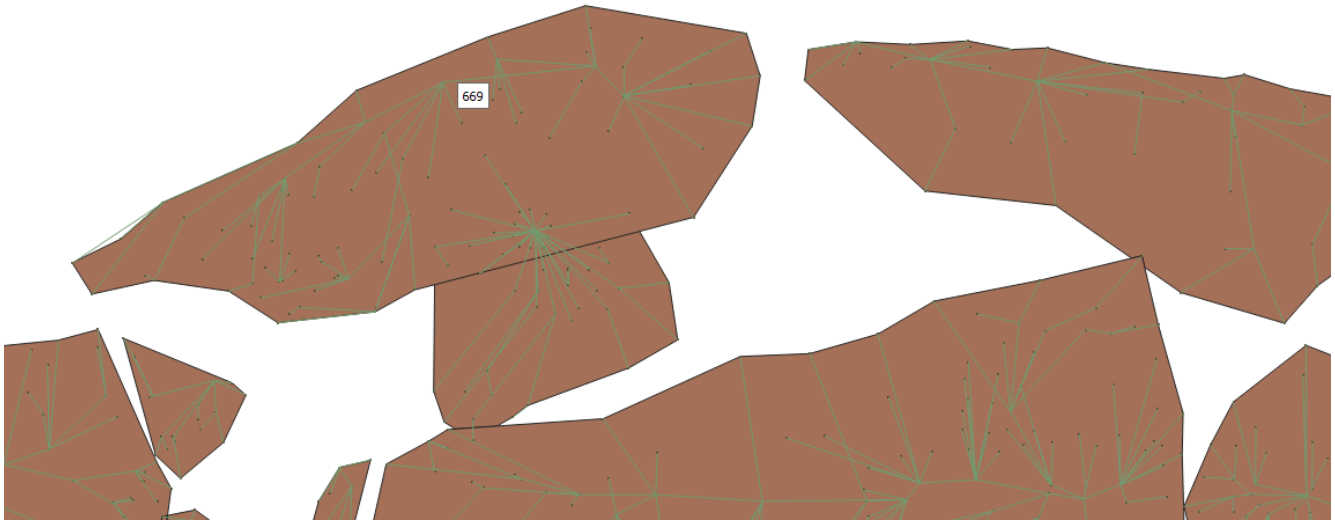
- File Before.png added

Debojyoti Mallick wrote:

> Debojyoti Mallick wrote:
>
> > Debojyoti Mallick wrote:
> >
> > > Applying Topology Rules to refine borders of Basins using QGIS.
> > >
> > > Used Check Geometry and Topology Checker to try and correct ends of basins.
> > >
> > > Some Basins looked good and followed the correct rule for overlap while a few of them did not.
> > >
> > > Checking through all the rules to eventually look at the possibility of removing the faulty overlaps automatically. Else will have to resort to manual editing for them.
>
> Check Geometry and Topology Checker cannot be used to correct overlaps as both of them define an overlap only considering how the boundaries of polygons are shared with one another.
> This defeats the purpose as the overlaps should be eliminated by the logic of the catchment points inside the basin and the direction of the drain to the lowest catchment area.
>
> Only using simple Topology checks results in such anomalies.



Before using Check Geometry



As you can see in the image above the overlaps are removed but they do not respect the direction of the drains nor do they respect the location of the catchment points.

Second alternative which was used was VClean using GRASS toolbox.

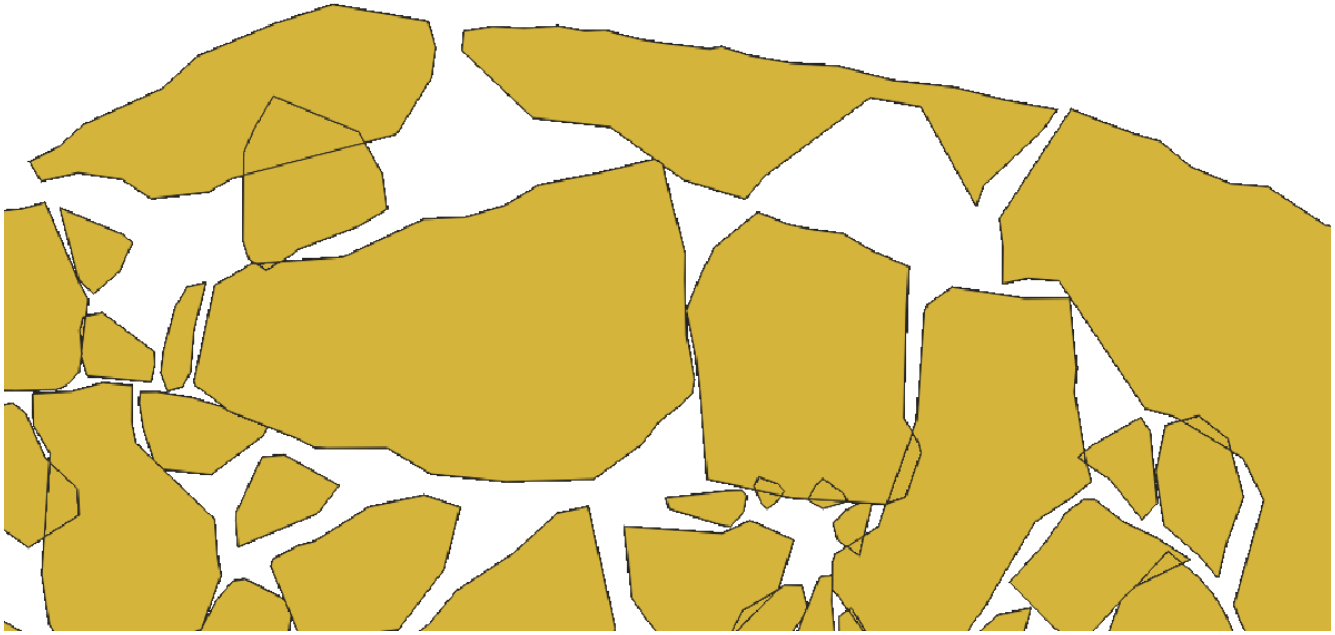
This uses the Topology Checker results to eventually clean the overlaps. This does give much better results but still creates some similar anomalies as discussed before. In addition to this it also automatically eliminates Silver Polygons which is not ideal as it would eventually eliminate the sub-basins and very small catchment areas.

To resolve geometric errors inclusive of Drains and Catchment points we did not use simple Topology rules as QGIS does not give the option of relations between points, lines and polygons.

Hence we use basic spatial concepts like Spatial Joins, Merge and Dissolve.

We first run a Spatial Join (Join by Location) between the Basins (Polygons) and the Drains (Lines)
A single attribute table with both results gives us more control.

The resulting layer is as follows.

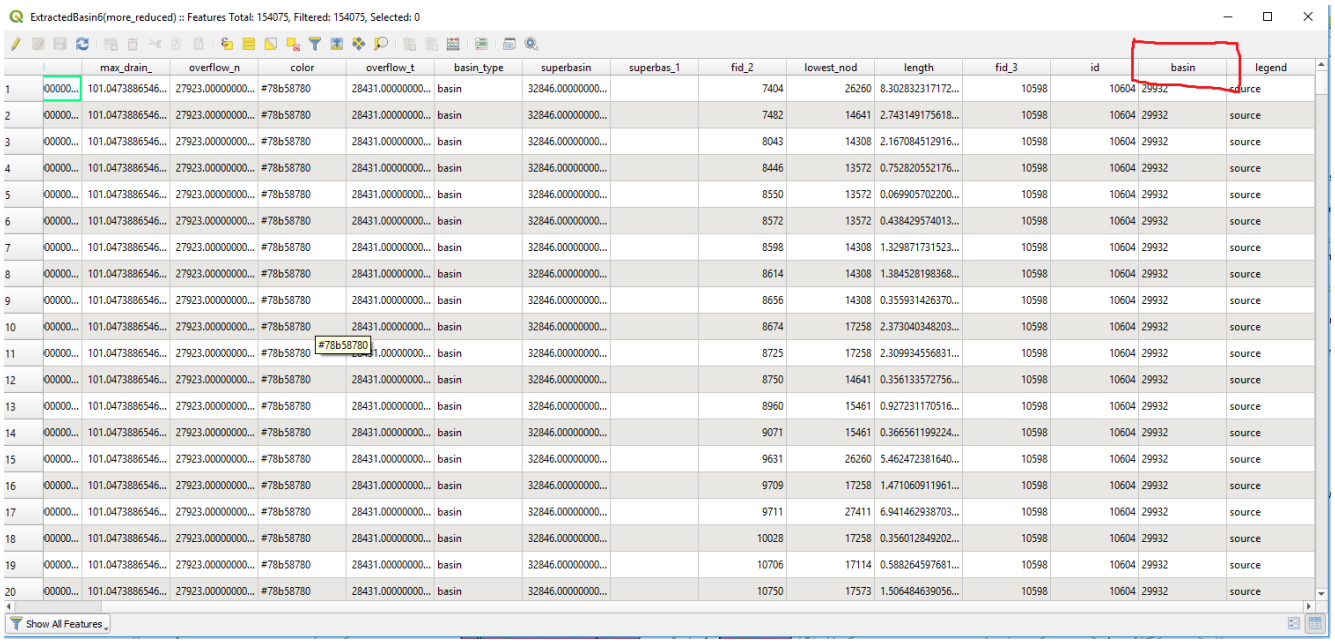


As the above figure shows, we can now see all the defined shapes of the basins and this will not show any overlaps.
This is now used as a base layer and we use the drain numbers to run our dissolve.

The resulting layer is not correct as consecutive drain numbers do not necessarily lead to or belong to the same basin.

Hence we have to include the catchment points and use those as a reference to define the boundaries for the basins.
On checking the attribute table the Basin column in the catchment points is a very reliable attribute to work with.

Hence ran a Spatial Join between the Basins, Drains and the Catchment Points.


ExtractedBasin6(more_reduced) :: Features Total: 154075, Filtered: 154075, Selected: 0

	max_drain_	overflow_n	color	overflow_t	basin_type	superbasin	superbas_1	fid_2	lowest_nod	length	fid_3	id	basin	legend
1	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		7404	26260	8.3028323171720...	10598	10604	29932	source
2	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		7482	14641	2.7431491756180...	10598	10604	29932	source
3	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8043	14308	2.1670845129160...	10598	10604	29932	source
4	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8446	13572	0.7528205521760...	10598	10604	29932	source
5	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8550	13572	0.0699057022000...	10598	10604	29932	source
6	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8572	13572	0.4384295740130...	10598	10604	29932	source
7	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8598	14308	1.3298717315230...	10598	10604	29932	source
8	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8614	14308	1.3845281983680...	10598	10604	29932	source
9	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8656	14308	0.3559314263700...	10598	10604	29932	source
10	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8674	17258	2.3730403482030...	10598	10604	29932	source
11	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8725	17258	2.3099345568310...	10598	10604	29932	source
12	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8750	14641	0.3561335727560...	10598	10604	29932	source
13	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		8960	15461	0.9272311705160...	10598	10604	29932	source
14	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9071	15461	0.3665611992240...	10598	10604	29932	source
15	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9631	26260	5.4624723816400...	10598	10604	29932	source
16	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9709	17258	1.4710609119610...	10598	10604	29932	source
17	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		9711	27411	6.9414629387030...	10598	10604	29932	source
18	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10028	17258	0.3560128492020...	10598	10604	29932	source
19	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10706	17114	0.5882645976810...	10598	10604	29932	source
20	00000...	101.0473886546...	27923.00000000...	#78b58780	28431.00000000...	basin	32846.00000000...		10750	17573	1.5064846390560...	10598	10604	29932	source

Show All Features

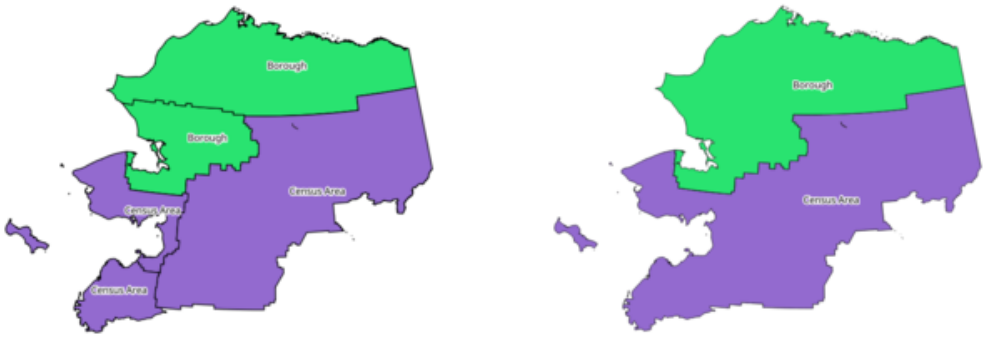
We now run GDAL Dissolve using the Catchment Points as a reference.

The results are good but we do have a few anomalies as the tool cannot decide the borders of the certain basins.

!Dissolve Problem1.png!

As per the image above, you will see that the polygons have been merged because the dissolve tool was not able to decide where the point belongs. But this is the only anomaly with regards to a single point.

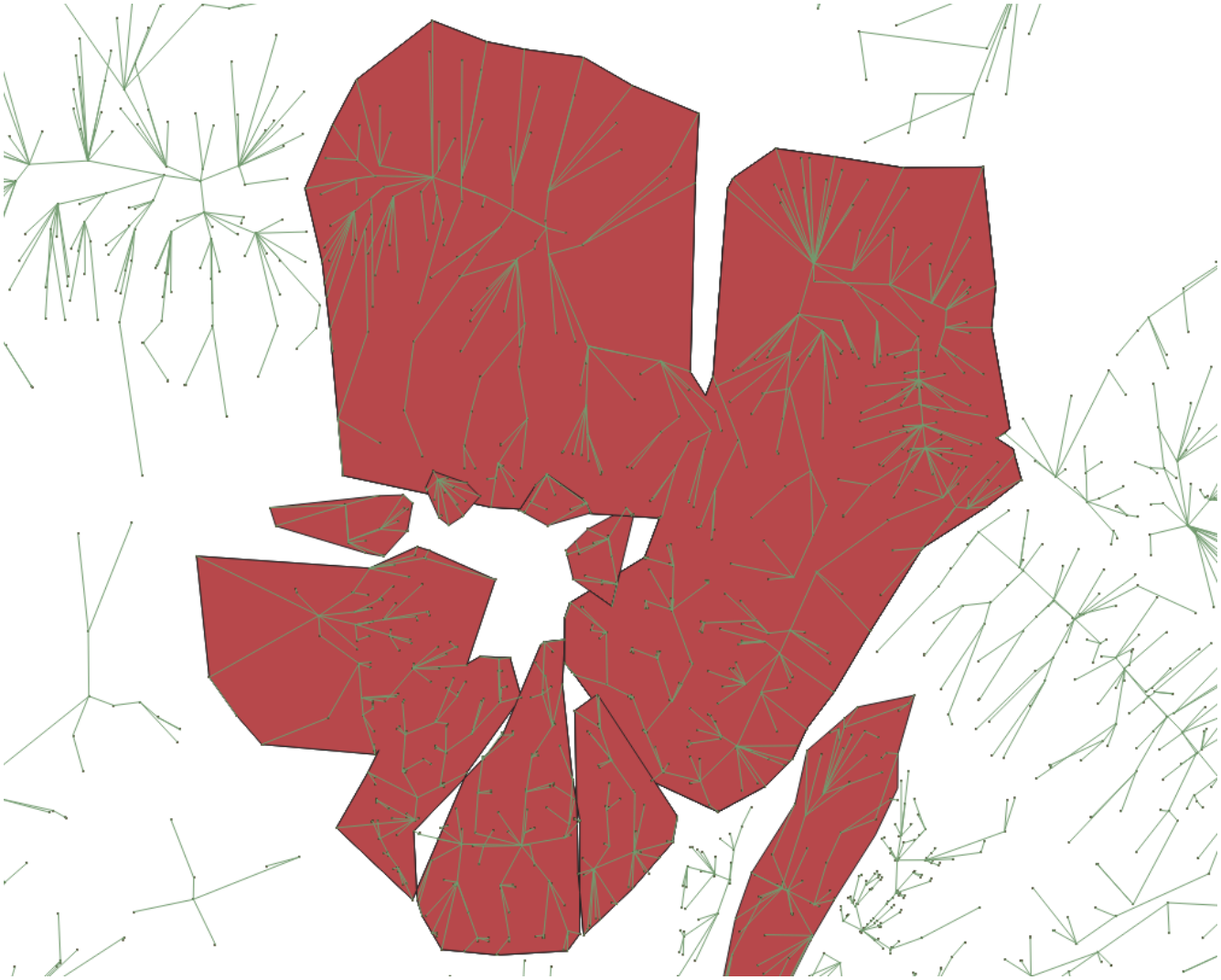
Keep in mind that we have considered the true definition and use of Dissolve illustrated below.



Dissolve the polygon layer on a common attribute

But the difference here is the fact that we are using the Catchment Points as the defining variable hence it should not merge the polygons rather define the borders of the basins (by eradicating unnecessary overlaps).

A similar issue of Merging Polygons occurs when there are multiple catchment points between two polygons.
The GDAL Dissolve tool is unable to decide the correct outline of the basin in such areas.



In the image above you can see another case of merged polygons.

The rest of the layer does not show this problem as 80% of the overlaps are eradicated correctly.

Hence we use Dissolve in Python/GeoPandas rather than use the GDAL tool in QGIS to check if the same issue is replicated.

There seems to be an issue with Jupyter with regards to reading files

gdf = gpd.read_file('SandBox/Debo/data/ExtractedBasin6more_reduced.shp')

Unfortunately files cannot be read. I will run this once it has been resolved.

Hence running it on VSCode to check the results. Takes time because of the hardware incompetence of my machine. Will be running this on a desktop to check the results.

Ran the dissolve using GeoPandas on Jupyter using a geopackage. Same results worried.png

This was the code used:

```
import geopandas
gdf = geopandas.read_file('data/Extractedbasin6_geopack.gpkg').to_crs({'init' :'epsg:4326'})

import numpy as np

gdf.plot(color='white', edgecolor='black')

import matplotlib.pyplot as plt
import matplotlib.lines as mlines
from matplotlib.colors import ListedColormap

gdf.geom_type.head()
```

Basin_bound = gdf['basin', 'geometry']

Basin_dissolve = Basin_bound.dissolve(by='basin')

Basin_dissolve

Basin_dissolve.plot(color='white', edgecolor='black')

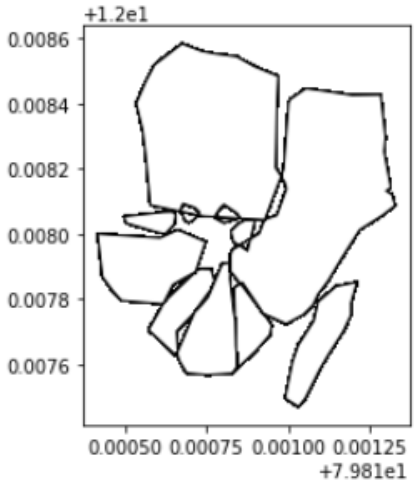
#Gives the same result

This concludes that there is no issue with the GDAL Dissolve and the logic used to get the output is flawed.

Will check for alternatives.

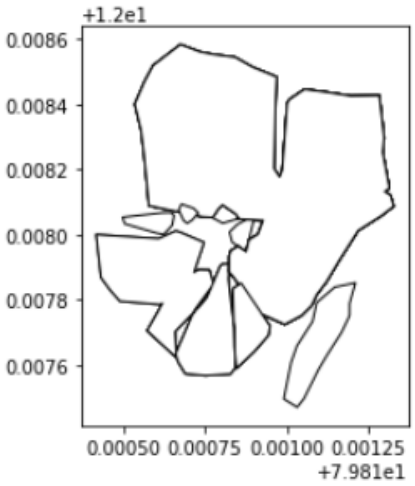
Just to give a clearer output with the code on Jupyter:

```
[4]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4f31796fd0>
```



Before

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4f31815518>
```



After running Dissolve

#4 - 07/06/2019 15:30 - Debojyoti Mallick

- Status changed from New to In Progress

#5 - 12/06/2019 16:43 - Debojyoti Mallick

- Status changed from In Progress to Feedback

- % Done changed from 0 to 100

Files

AfterCheckGeometry.png	107 KB	07/06/2019	Debojyoti Mallick
AfterDissolve2.png	242 KB	07/06/2019	Debojyoti Mallick
AttributeforDissolve.png	95.8 KB	07/06/2019	Debojyoti Mallick
BeforeCheckGeometry.png	98.9 KB	07/06/2019	Debojyoti Mallick
BeforeDissolve2.png	100 KB	07/06/2019	Debojyoti Mallick
Dissolve Problem1.png	192 KB	07/06/2019	Debojyoti Mallick
JoinwithDrain.png	68.7 KB	07/06/2019	Debojyoti Mallick
Dissolve.png	38.3 KB	07/06/2019	Debojyoti Mallick
After.png	29.3 KB	07/06/2019	Debojyoti Mallick
Before.png	31.4 KB	07/06/2019	Debojyoti Mallick