alpha2_resistance_connectivity

September 2, 2016

I wanted to examine how α_2, which we think of as modulating the resistance of the landscape, affects the resulting potential connectivity surface. Since potential connectivity is given by:

$$C^P(i) = \sum_j Pr(g[i,j]) = \sum_j e^{-\alpha_1 d_{ecol}(i,j)^2}$$

I break up the $d_{ecol}(i,j)$ term into the part that depends on α_2, which can be factored out, and the part that depends only on the least cost path.

$$d_{ecol}(i,j) = min_{\mathcal{L}} \sum_{p=1}^{m+1} cost(v_p, v_{p+1}) d_{euc}(v_p, v_{p+1}) = min_{\mathcal{L}} \sum_{p=1}^{m+1} e^{\alpha_2 \frac{z(v_p)+z(v_{p+1})}{2}} d_{euc}(v_p, v_{p+1}) = min_{\mathcal{L}} \sum_{p=1}^{m+1} e^{\alpha_2} e^{\frac{z(v_p)+z(}{2}}$$

Now, I can treat the *blue* term as distance between pixels (or the least cost path between pixels without the modulating effect of α_2), which I vary from 0 to 2 units. (Note, I do not know what order of magnitude these distances actually are; but the values I get for $p_{i,j}$ are comparable to the values in the potmat matrices I am working with.)

To get at potential connectivity, set $p_0 = 1$, use either 0.25 or 1.75 for α_2, and compute α_1 based on the effective sigma. I used the effective sigma values corresponding to these α_2 taken from the appendix for Dana's manuscript. Larger α_2 uses a larger σ with the intention of keeping home range size roughly the same.

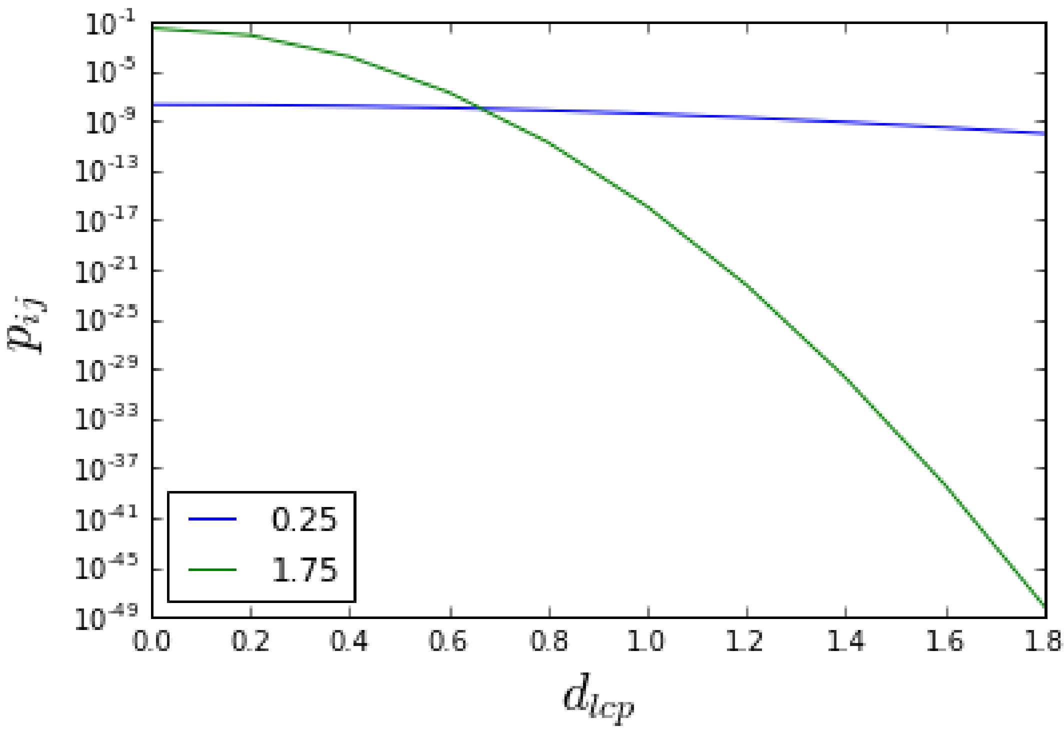
$p_{i,j} = p_0 e^{-\alpha_1 d_{ecol}(i,j)^2} = e^{-\alpha_1} e^{-d_{ecol}(i,j)^2} = e^{-\frac{1}{2\sigma^2}} e^{-d_{ecol}(i,j)^2} = e^{-\frac{1}{2\sigma^2}} e^{-e^{2\alpha_2}} e^{-d_{lcp}(i,j)^2}$

```
In [1]: import math
        import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline

        d_lcp = np.arange(0,2,0.2)
        # print(d_lcp)
        alpha0 = 1
        alpha2 = [0.25, 1.75]
        effsig = [0.1677509, 0.3749552]
        alpha1 = [0]*len(alpha2)
        for idx in range(len(alpha2)):
            alpha1[idx] = 1/(2*effsig[idx]**2)
        p_ij = dict()
```

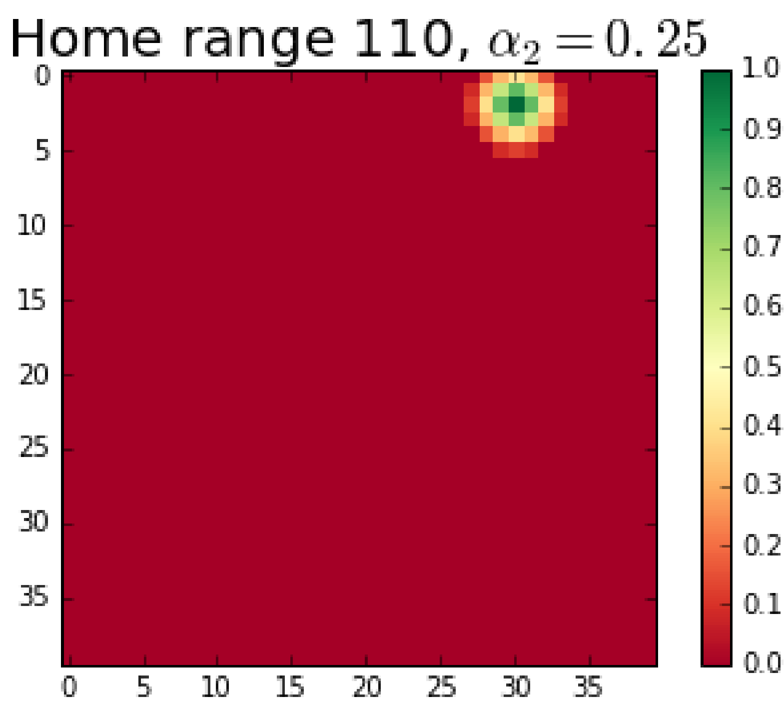
```
for a in range(len(alpha2)):
    a1 = alpha1[a]
    a2 = alpha2[a]
    p_ij[a2] = list()
    for d in range(len(d_lcp)):
        d_ecol = math.exp(a2)*float(d_lcp[d])
        pij = alpha0*math.exp(-a1)*math.exp(-(d_ecol*d_ecol))
        p_ij[a2].append(pij)
    plt.semilogy(d_lcp, p_ij[a2])
plt.ylabel(r'$p_{ij}$', fontsize=20)
plt.xlabel(r'$d_{lcp}$', fontsize=20)
plt.legend(alpha2,loc='lower left')
plt.show()
```

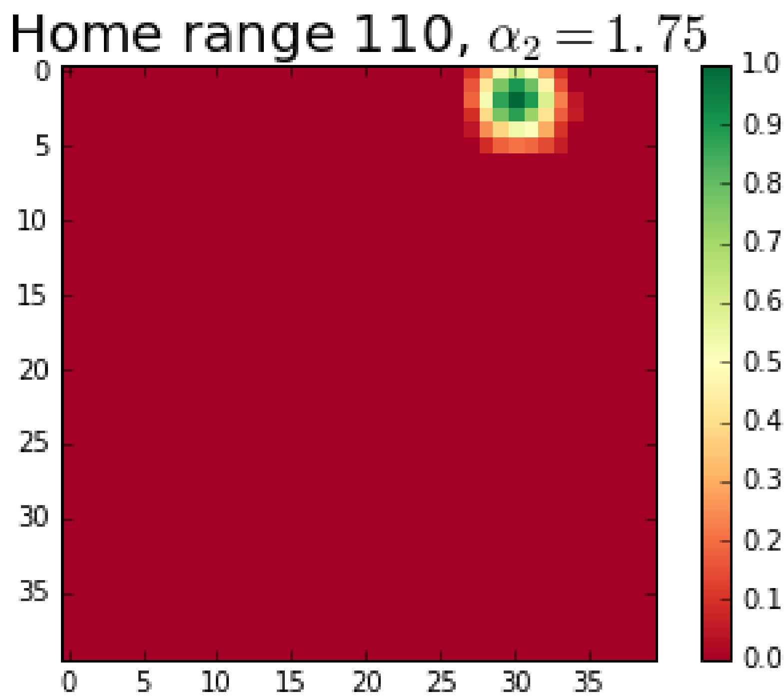
```
/usr/local/lib/python2.7/site-packages/matplotlib/font_manager.py:273: UserWarning:
  warnings.warn('Matplotlib is building the font cache using fc-list. This may take
```



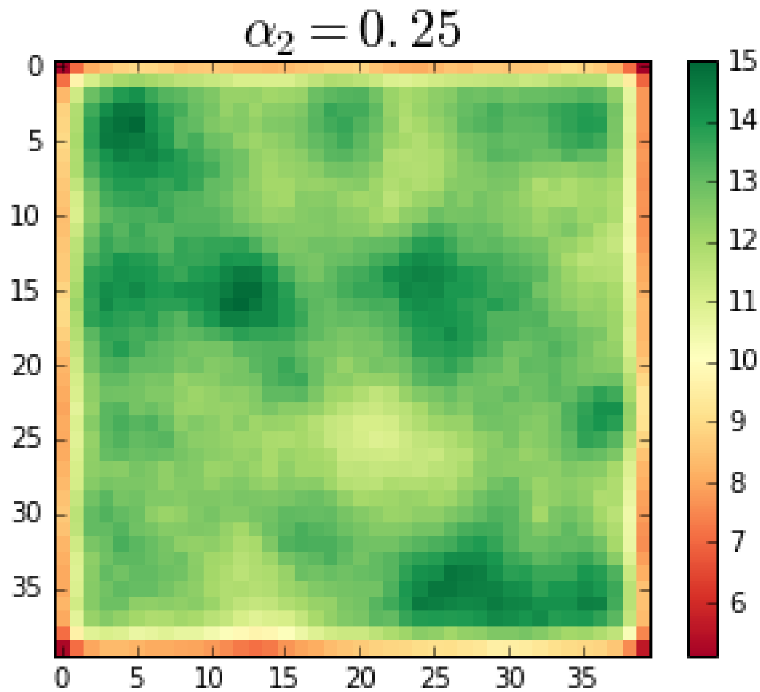
For smaller d_{lcp} values, the high resistance surface actually has a higher use probability, giving rise to a larger potential connectivity. Seen another way, if we have a fixed 95% home range, in the $\alpha_2 = 1.75$ case the use probabilities of those pixels is higher than for the low-resistance case, as can be seen by the darker green coloring (higher use probabilities) in $\alpha_2 = 1.75$ of the 8 pixels immediately surrounding activity center 110.

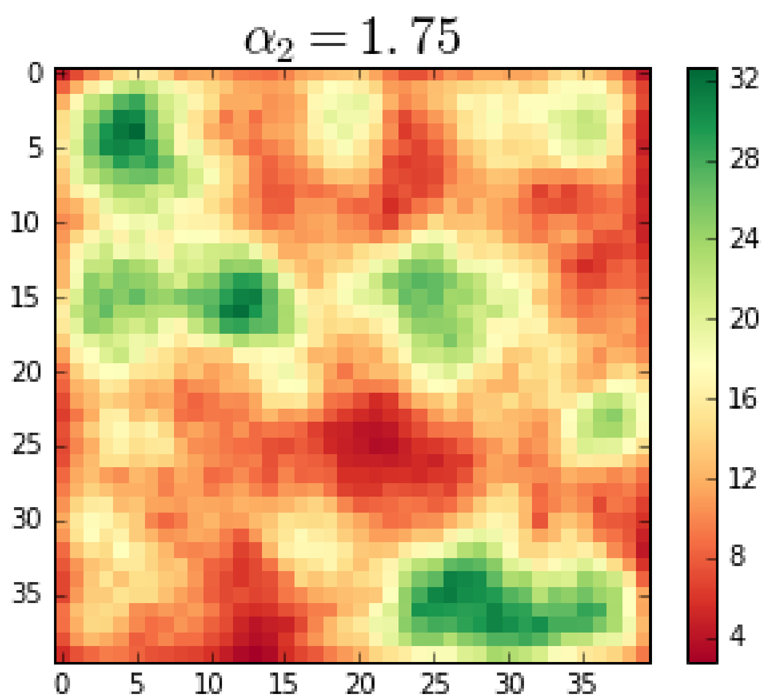
In [21]:
```
import rasterio
import matplotlib.cm as cm
datasetfilename = "../Desktop/hropt/Data/simcov_a2025_S100_useprobs.txt"
with open(datasetfilename) as potmatdata:
    potmat = np.loadtxt(potmatdata)
    low = potmat < 0.05
    potmat[low] = 0
    hr110 = potmat[110,].reshape(40,40)
    plt.imshow(hr110, interpolation='nearest', cmap=cm.RdYlGn, alpha=1.0)
    plt.colorbar()
    plt.title(r'Home range 110, $\alpha_2=0.25$', fontsize=20)
    plt.show()
datasetfilename = "../Desktop/hropt/Data/simcov_a2175_S100_useprobs.txt"
with open(datasetfilename) as potmatdata:
    potmat = np.loadtxt(potmatdata)
    low = potmat < 0.05
    potmat[low] = 0
    hr110 = potmat[110,].reshape(40,40)
    plt.imshow(hr110, interpolation='nearest', cmap=cm.RdYlGn, alpha=1.0)
    plt.colorbar()
    plt.title(r'Home range 110, $\alpha_2=1.75$', fontsize=20)
    plt.show()
```





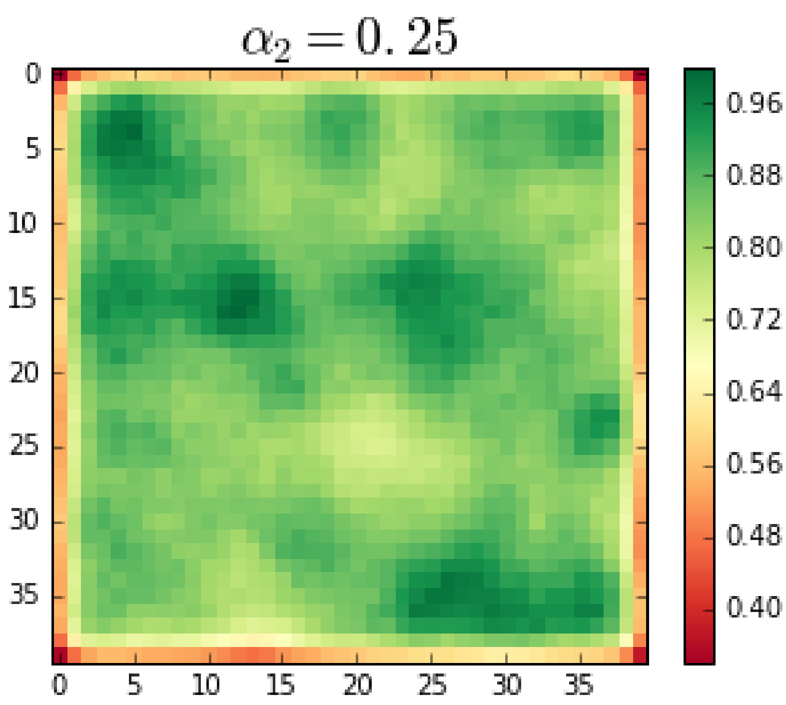

```
In [2]: datasetfilename = "../Desktop/hropt/Data/simcov_a2025_S100_pc.tif"
        with rasterio.open(datasetfilename) as src:
            r = src.read()
            data = r.squeeze()
            plt.figure(figsize=(5, 4))
            plt.imshow(data, interpolation='nearest', cmap=cm.RdYlGn, alpha=1.0) #
            plt.colorbar()
            plt.grid(False)
            plt.title(r'$\alpha_2=0.25$', fontsize=20)
            plt.show()
        datasetfilename = "../Desktop/hropt/Data/simcov_a2175_S100_pc.tif"
        with rasterio.open(datasetfilename) as src:
            r = src.read()
            data = r.squeeze()
            plt.figure(figsize=(5, 4))
            plt.imshow(data, interpolation='nearest', cmap=cm.RdYlGn, alpha=1.0) #
            plt.colorbar()
            plt.grid(False)
            plt.title(r'$\alpha_2=1.75$', fontsize=20)
            plt.show()
```

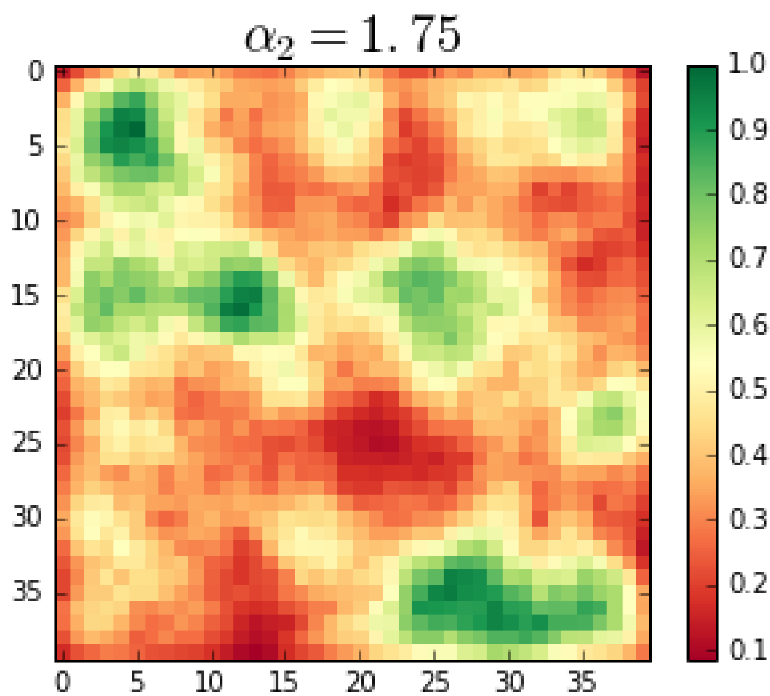




The two landscapes are identical except for the α_2 parameter. The patches of "relatively good" One possible way to handle this might be to rescale all the values in potential connectivity:

In [14]:
```
datasetfilename = "../Desktop/hropt/Data/simcov_a2025_S100_pc.tif"
with rasterio.open(datasetfilename) as src:
    r = src.read()
    data = r.squeeze()
    data = data/data.max()
    plt.figure(figsize=(5, 4))
    plt.imshow(data, interpolation='nearest', cmap=cm.RdYlGn, alpha=1.0) #
    plt.colorbar()
    plt.grid(False)
    plt.title(r'$\alpha_2=0.25$', fontsize=20)
    plt.show()
datasetfilename = "../Desktop/hropt/Data/simcov_a2175_S100_pc.tif"
with rasterio.open(datasetfilename) as src:
    r = src.read()
    data = r.squeeze()
    data = data/data.max()
    plt.figure(figsize=(5, 4))
    plt.imshow(data, interpolation='nearest', cmap=cm.RdYlGn, alpha=1.0) #
    plt.colorbar()
    plt.grid(False)
    plt.title(r'$\alpha_2=1.75$', fontsize=20)
    plt.show()
```




$\alpha_2 = 1.75$
0
5
10
15
20
25
30
35
0
5
10
15
20
25
30
35
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1

```
In [ ]:
```