SF-L: Seattle Fault Mw 7.3?

Two slightly different versions of this fault are given in

NOAA Technical Memorandum OAR PMEL-124 (https://repository.library.noaa.gov/view/noaa/11033): NOAA TIME SEATTLE TSUNAMI MAPPING PROJECT: PROCEDURES, DATA SOURCES, AND PRODUCTS, by Vasily V. Titov, Frank I. González, Harold O. Mofjeld, and Angie J. Venturato

NOAA Technical Memorandum OAR PMEL-147 (https://repository.library.noaa.gov/view/noaa/11189): MODELING TSUNAMI INUNDATION FOR HAZARD MAPPING AT EVERETT, WASHINGTON, FROM THE SEATTLE FAULT, by C. Chamberlin and D. Arcas

Table 1 of PMEL-124 gives the fault width 20km for all subfaults, whereas Table 1 of PMEL-147 gives these as 35km.

Neither report lists the rake or the depth of the subfaults. The rigidity (shear modulus) mu used in calculating the moment magitude is also not provided.

PMEL-124 says "The lower depth of the fault is also a subject of active debate. Different hypotheses put the low boundary of the fault anywhere from 12 to 28 km deep." Since the dip is 60 degrees, with a width of 20km the fault top is 17.3 km above the fault bottom, so the fault bottom cannot be shallower than this.

The csv file provided by Frank has the width set to 35km, the rake set to 90 degrees, and rigidity to 3e10 Pa. This gives Mw 7.55.

Experiments with different fault parameters

If the widths are reduced to 20km, then the magnitude is Mw 7.39. This can be decreased by decreasing mu.

Setting the rake to 90 degrees is appropriate for megathrust events on the subduction zone, but probably not for the Seattle fault?

In the experiments below, the uplift or subsidence has been estimated at three points: Alki Point, Restoration Point, and West Point, taken from Table 2 of PMEL-124:

Displacements	Model	Observations
Alki Point	3.9m	4m
Restoration Point	7.1m	7m
West Point	-1.3m	–1 ± 0.5 m

In [1]:
```
from __future__ import absolute_import
from __future__ import print_function
```

In [2]:
```
%pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

In [41]:
```
import os,sys
import numpy
#sys.path.append('/Users/rjl/git/WA_EMD_Snohomish/geoclaw')
#import topotools
from clawpack.geoclaw import dtopotools, topotools
from copy import deepcopy
```

In [4]:
```
pmel_124_SFdeform = imread('pmel_124_SFdeform.png')
```

In [5]:
```
topodir = '/Users/rjl/git/WA_EMD_Snohomish//topo/topofiles'
topo1 = topotools.Topography()
topo1.read(topodir+'/PT_2sec.asc', topo_type=3)
topo2 = topotools.Topography()
topo2.read(topodir+'/juandefuca_2sec_mhw.asc', topo_type=3)
topo3 = topotools.Topography()
topo3.read(topodir+'/pugetsound_2sec_mhw_llcorner.tt3', topo_type=3)
```

In [6]:
```
def eval_dtopo(x,y, dtopo):
    i = find(dtopo.x<=x).max()
    j = find(dtopo.y<=y).max()
    Xi = dtopo.X[j,i]
    Yj = dtopo.Y[j,i]
    dz = dtopo.dZ[-1,j,i]
    #print("At x = %6.3f, y = %6.3f:   dz = %6.3f" % (Xi,Yj,dz))
    return Xi,Yj,dz
```

Read in csv file, print subfault info:

In [42]:
```
defaults = {}
coordinate_specification = 'top center'
input_units = {'slip': 'm', 'depth': 'km',
               'length': 'km', 'width': 'km'}
rupture_type = 'static'
subfault_fname = 'SF_Mw7.3_W35_mu30.csv'
fault = dtopotools.CSVFault()
fault.read(subfault_fname, input_units, coordinate_specification,
           rupture_type)
fault_original = deepcopy(fault)
print(fault.Mw())
```

```
7.54814143444
```

In [8]:
```
for s in fault.subfaults:
    print(s)
```

```
Subfault Characteristics:
  Coordinates: (-122.7599344, 47.6115777) (top center)
  Dimensions (L,W): (15200.0, 35000.0) m
  Depth: 500.0 m
  Rake, Strike, Dip: 90.0, 87.9, 60.0
  Slip, Moment: 1.0 m, 1.596e+19 N-m
  Fault Centroid: [-122.75565480587498, 47.532896684064049, 15655.44456
6227676]

Subfault Characteristics:
  Coordinates: (-122.6165584, 47.6157655) (top center)
  Dimensions (L,W): (6300.0, 35000.0) m
  Depth: 500.0 m
  Rake, Strike, Dip: 90.0, 86.6, 60.0
  Slip, Moment: 1.0 m, 6.615e+18 N-m
  Fault Centroid: [-122.60963149376214, 47.537170191092798, 15655.44456
6227676]

Subfault Characteristics:
  Coordinates: (-122.5154909, 47.6132604) (top center)
  Dimensions (L,W): (8900.0, 35000.0) m
  Depth: 500.0 m
  Rake, Strike, Dip: 90.0, 96.0, 60.0
  Slip, Moment: 12.0 m, 1.1214e+20 N-m
  Fault Centroid: [-122.52769910182411, 47.534957818400926, 15655.44456
6227676]

Subfault Characteristics:
  Coordinates: (-122.4397627, 47.6000508) (top center)
  Dimensions (L,W): (3200.0, 35000.0) m
  Depth: 500.0 m
  Rake, Strike, Dip: 90.0, 128.8, 60.0
  Slip, Moment: 11.0 m, 3.696e+19 N-m
  Fault Centroid: [-122.51292721334609, 47.538690487193911, 15655.44456
6227676]

Subfault Characteristics:
  Coordinates: (-122.3474066, 47.5826645) (top center)
  Dimensions (L,W): (11500.0, 35000.0) m
  Depth: 500.0 m
  Rake, Strike, Dip: 90.0, 99.3, 60.0
  Slip, Moment: 4.0 m, 4.83e+19 N-m
  Fault Centroid: [-122.36626977531962, 47.504965506523504, 15655.44456
6227676]

Subfault Characteristics:
  Coordinates: (-122.1735094, 47.5847905) (top center)
  Dimensions (L,W): (14900.0, 35000.0) m
  Depth: 500.0 m
  Rake, Strike, Dip: 90.0, 81.0, 60.0
  Slip, Moment: 1.0 m, 1.5645e+19 N-m
  Fault Centroid: [-122.15524887518968, 47.507025950778818, 15655.44456
6227676]
```

In [9]:
```
print("There are %s subfaults" % len(fault.subfaults))
widths = [s.width for s in fault.subfaults]
print("subfault widths: ", widths)
print("Mw = %5.2f" % fault.Mw())
```

```
There are 6 subfaults
subfault widths:  [35000.0, 35000.0, 35000.0, 35000.0, 35000.0, 35000.
0]
Mw =  7.55
```

Create deformation (dtopo):

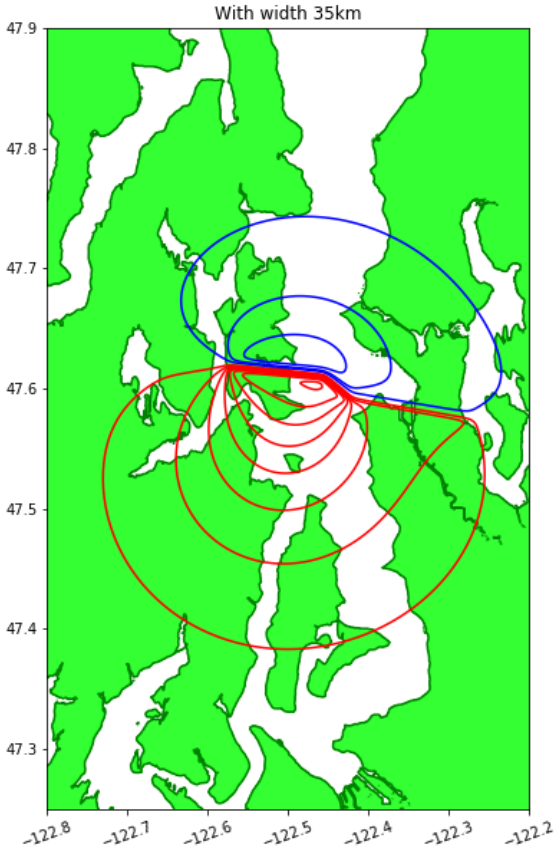
In [10]:
```
x = numpy.arange(-123, -122.2, 2./3600.)
y = numpy.arange(47., 48.6, 2./3600.)
times = [1.]

fault.create_dtopography(x,y,times)
dtopo = fault.dtopo
print('dtopo min, max: ',dtopo.dZ.min(),dtopo.dZ.max())
```

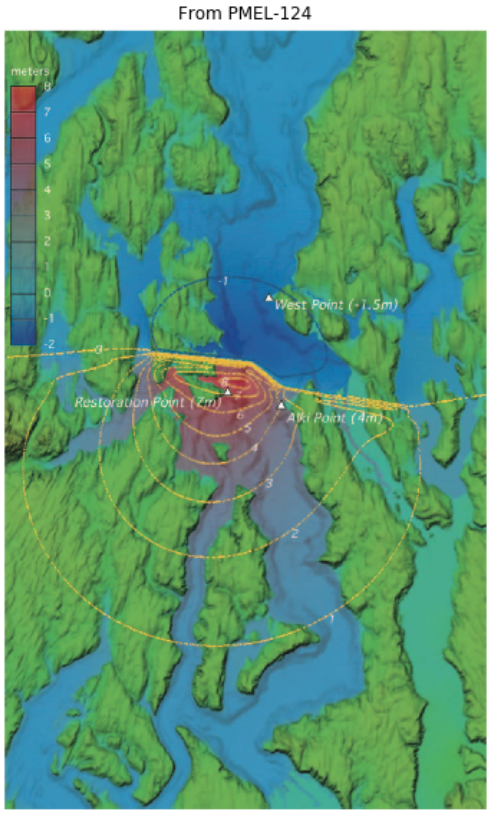
```
dtopo min, max:  -1.78449303433 8.15784108738
```

In [11]:
```
dtopo_original = deepcopy(dtopo)
```

In [47]:
```
figure(figsize=(15,10))
clf()
subplot(1,2,1)
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo_original min, max: ',dtopo_original.dZ.min(),dtopo_original
.dZ.max())

clines = arange(-2,0,0.5)
print('blue clines = ',clines)
CB = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-1
,:,:], clines,
        colors='b', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)
CR = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-1
,:,:], clines,
        colors='r', linestyles='-')
#clabel(CR,clines[::3])

ticklabel_format(format='plain',useOffset=False)

#xticks(arange(-122.6,-122.14,.06))
#yticks(arange(47.8,48.35,.04))
xticks(rotation=20,fontsize=10)
yticks(fontsize=10)
gca().set_aspect(1./cos(48*pi/180.))
title('With width 35km')

subplot(1,2,2)
imshow(pmel_124_SFdeform)
axis('off')
title('From PMEL-124')

savefig('SFL-PMEL.png', bbox_inches='tight')
```

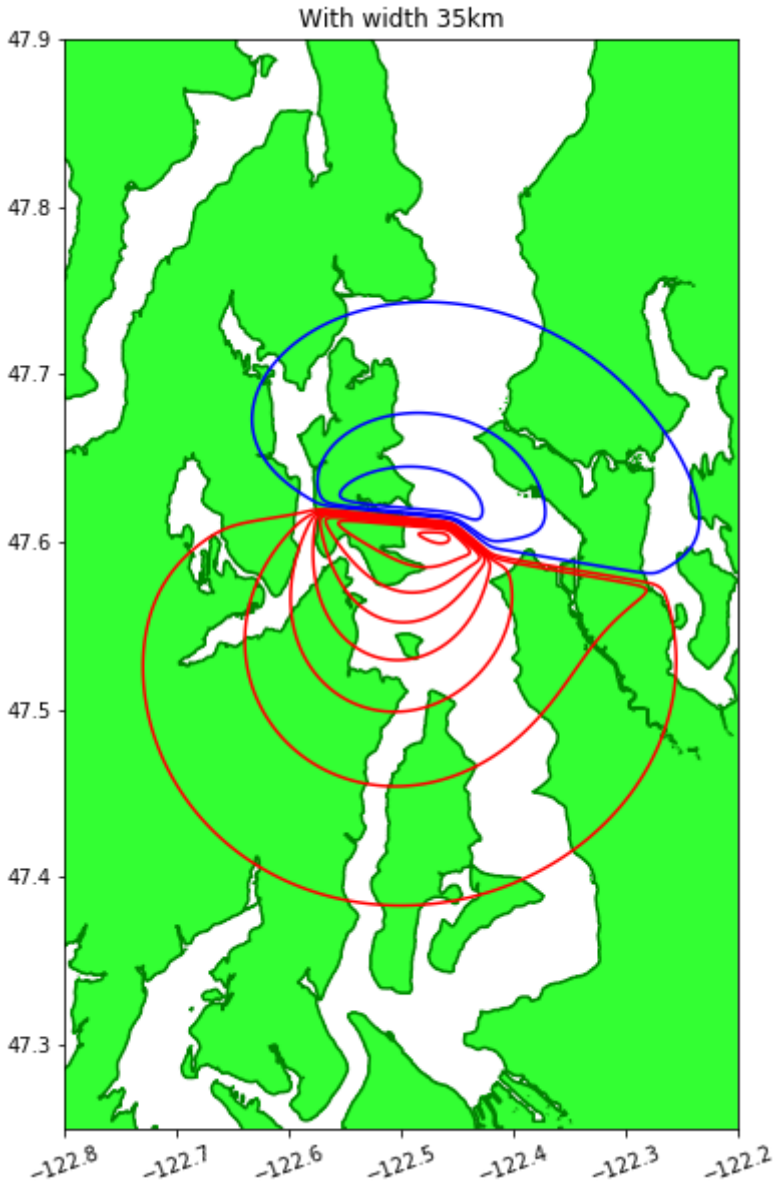
```
dtopo_original min, max:  -1.78449303433 8.15784108738
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
```





In [71]:
```
figure(figsize=(15,10))
clf()
subplot(1,2,1)
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo_original min, max: ',dtopo_original.dZ.min(),dtopo_original
.dZ.max())

clines = arange(-2,0,0.5)
print('blue clines = ',clines)
CB = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-1
,:,:], clines,
        colors='b', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)
CR = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-1
,:,:], clines,
        colors='r', linestyles='-')
#clabel(CR,clines[::3])

ticklabel_format(format='plain',useOffset=False)

#xticks(arange(-122.6,-122.14,.06))
#yticks(arange(47.8,48.35,.04))
xticks(rotation=20,fontsize=10)
yticks(fontsize=10)
gca().set_aspect(1./cos(48*pi/180.))
title('With width 35km')

savefig('SFL-csv35.png', bbox_inches='tight')
```

```
dtopo_original min, max:  -1.78449303433 8.15784108738
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
```



Estimate dtopo at three points to compare with PMEL-124:

Displacements	Model	Observations
Alki Point	3.9m	4m
Restoration Point	7.1m	7m
West Point	-1.3m	–1 ± 0.5 m

In [13]:
```
pts = [('Alki Point',-122.421,47.575),
       ('Restoration Point',-122.481,47.584),
       ('West Point',-122.436,47.662)]

for pt in pts:
    Xi,Yj,dz = eval_dtopo(pt[1],pt[2],dtopo_original)
    print('%s: x = %7.3f, y = %7.3f, dz = %7.3f' % (pt[0].rjust(18), Xi,
Yj,dz))
```

```
        Alki Point: x = -122.421, y =  47.574, dz =   3.756
 Restoration Point: x = -122.481, y =  47.584, dz =   6.928
        West Point: x = -122.436, y =  47.662, dz =  -1.098
```

Adjust subfault widths to agree with PMEL-124:

In [72]:
```
fault = deepcopy(fault_original)
for s in fault.subfaults:
    s.width = 20e3
    s.calculate_geometry()
widths = [s.width for s in fault.subfaults]
print("subfault widths: ", widths)
print("Mw = %5.2f" % fault.Mw())
```

```
subfault widths:  [20000.0, 20000.0, 20000.0, 20000.0, 20000.0, 20000.
0]
Mw =  7.39
```

In [73]:
```
x = numpy.arange(-123, -122.2, 2./3600.)
y = numpy.arange(47., 48.6, 2./3600.)
times = [1.]

fault.create_dtopography(x,y,times)
dtopo = fault.dtopo
print('dtopo min, max: ',dtopo.dZ.min(),dtopo.dZ.max())
```

```
dtopo min, max:  -1.79552620623 8.04992536327
```

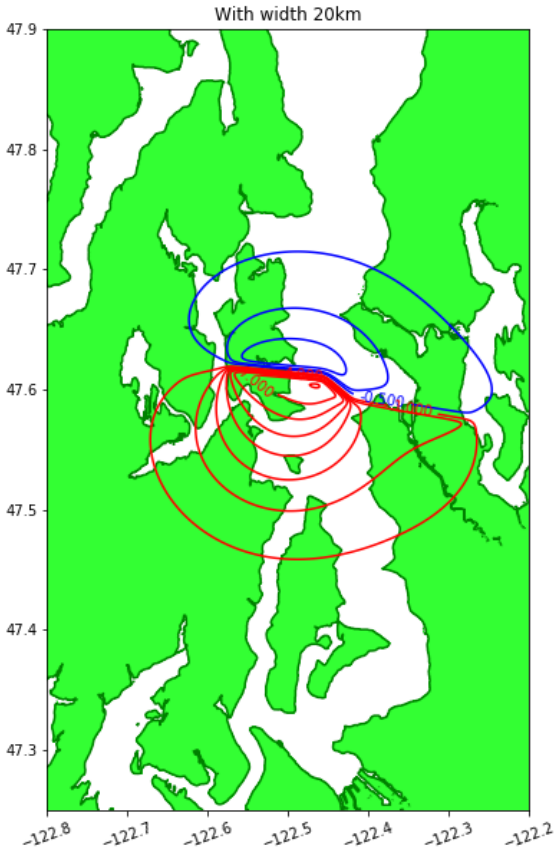
In [74]:
```
pts = [('Alki Point',-122.421,47.575),
       ('Restoration Point',-122.481,47.584),
       ('West Point',-122.436,47.662)]

for pt in pts:
    Xi,Yj,dz = eval_dtopo(pt[1],pt[2],dtopo)
    print('%s: x = %7.3f, y = %7.3f, dz = %7.3f' % (pt[0].rjust(18), Xi,
Yj,dz))
```

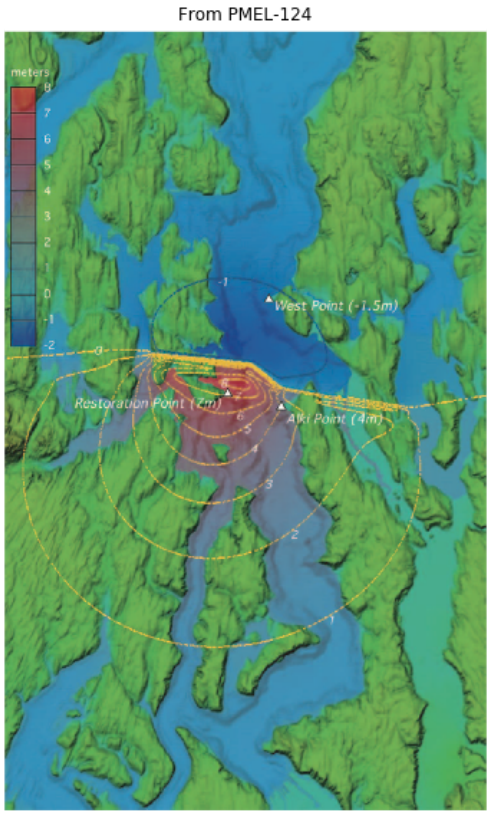
```
        Alki Point: x = -122.421, y =  47.574, dz =   3.517
 Restoration Point: x = -122.481, y =  47.584, dz =   6.663
        West Point: x = -122.436, y =  47.662, dz =  -0.973
```

In [75]:
```
figure(figsize=(15,10))
clf()
subplot(1,2,1)
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo min, max: ',dtopo.dZ.min(),dtopo.dZ.max())

clines = arange(-2,0,0.5)
print('blue clines = ',clines)
CB = contour(dtopo.X, dtopo.Y, dtopo.dZ[-1,:,:], clines,
        colors='b', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)
CR = contour(dtopo.X, dtopo.Y, dtopo.dZ[-1,:,:], clines,
        colors='r', linestyles='-')
#clabel(CR,clines[::3])

ticklabel_format(format='plain',useOffset=False)

#xticks(arange(-122.6,-122.14,.06))
#yticks(arange(47.8,48.35,.04))
xticks(rotation=20,fontsize=10)
yticks(fontsize=10)
gca().set_aspect(1./cos(48*pi/180.))
title('With width 20km')

subplot(1,2,2)
imshow(pmel_124_SFdeform)
axis('off')
title('From PMEL-124')
```

```
dtopo min, max:  -1.79552620623 8.04992536327
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
```

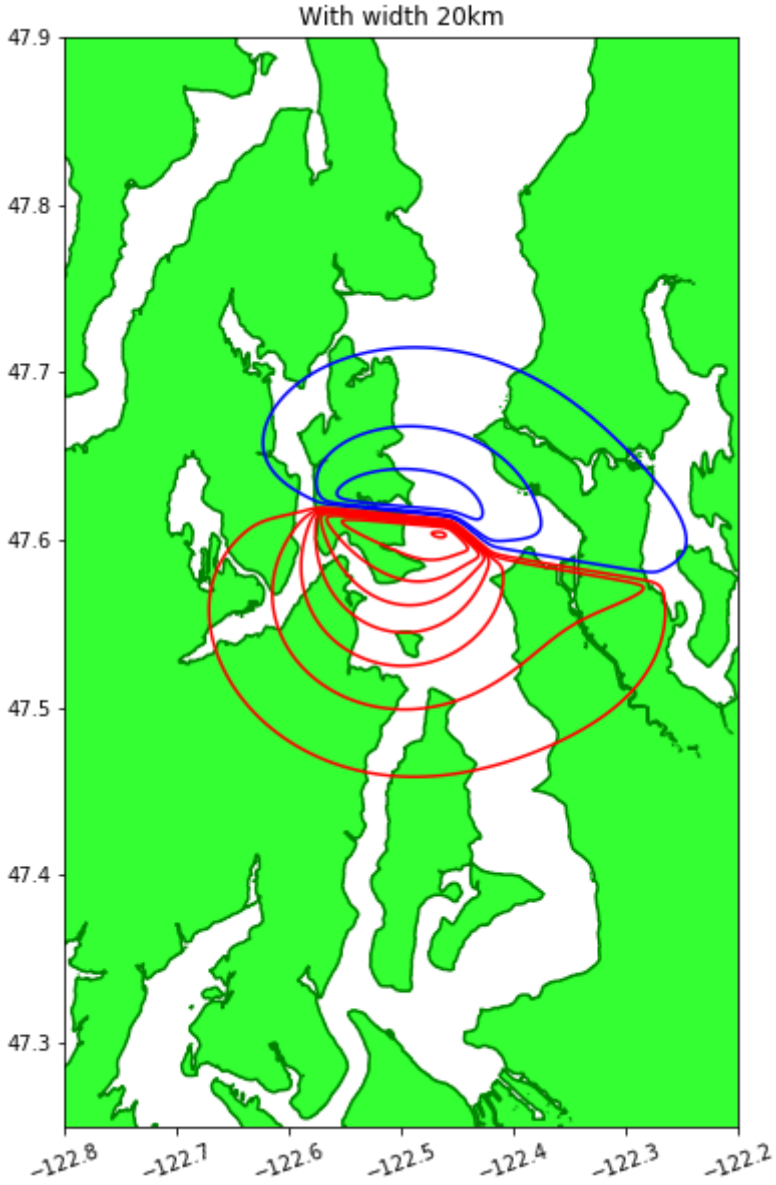
Out[75]: <matplotlib.text.Text at 0x11bd22828>





In [76]:
```
figure(figsize=(15,10))
clf()
subplot(1,2,1)
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo min, max: ',dtopo.dZ.min(),dtopo.dZ.max())

clines = arange(-2,0,0.5)
print('blue clines = ',clines)
CB = contour(dtopo.X, dtopo.Y, dtopo.dZ[-1,:,:], clines,
        colors='b', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)
CR = contour(dtopo.X, dtopo.Y, dtopo.dZ[-1,:,:], clines,
        colors='r', linestyles='-')
#clabel(CR,clines[::3])

ticklabel_format(format='plain',useOffset=False)

#xticks(arange(-122.6,-122.14,.06))
#yticks(arange(47.8,48.35,.04))
xticks(rotation=20,fontsize=10)
yticks(fontsize=10)
gca().set_aspect(1./cos(48*pi/180.))
title('With width 20km')

savefig('SFL-csv20.png',bbox_inches='tight')
```

```
dtopo min, max:  -1.79552620623 8.04992536327
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
```



Experiment with different depths:

In [18]:
```
fault = deepcopy(fault_original)
depths = [s.depth for s in fault.subfaults]
print("subfault depths: ", depths)
```

```
subfault depths:  [500.0, 500.0, 500.0, 500.0, 500.0, 500.0]
```

In [19]:
```
for s in fault.subfaults:
    s.depth = 100.
depths = [s.depth for s in fault.subfaults]
for s in fault.subfaults:
    s.calculate_geometry()
print("subfault depths: ", depths)
print("Mw = %5.2f" % fault.Mw())

fault.create_dtopography(x,y,times)
dtopo = fault.dtopo

for pt in pts:
    Xi,Yj,dz = eval_dtopo(pt[1],pt[2], dtopo)
    print('%s: x = %7.3f, y = %7.3f, dz = %7.3f' % (pt[0].rjust(18), Xi,
Yj,dz))
```

```
subfault depths:  [100.0, 100.0, 100.0, 100.0, 100.0, 100.0]
Mw =  7.39
        Alki Point: x = -122.421, y =  47.574, dz =   3.357
 Restoration Point: x = -122.481, y =  47.584, dz =   6.895
        West Point: x = -122.436, y =  47.662, dz =  -0.934
```

In [20]:
```
for s in fault.subfaults:
    s.depth = 1000.
depths = [s.depth for s in fault.subfaults]
for s in fault.subfaults:
    s.calculate_geometry()
print("subfault depths: ", depths)
print("Mw = %5.2f" % fault.Mw())

fault.create_dtopography(x,y,times)
dtopo = fault.dtopo

for pt in pts:
    Xi,Yj,dz = eval_dtopo(pt[1],pt[2], dtopo)
    print('%s: x = %7.3f, y = %7.3f, dz = %7.3f' % (pt[0].rjust(18), Xi,
Yj,dz))
```

```
subfault depths:  [1000.0, 1000.0, 1000.0, 1000.0, 1000.0, 1000.0]
Mw =  7.39
        Alki Point: x = -122.421, y =  47.574, dz =   3.658
 Restoration Point: x = -122.481, y =  47.584, dz =   6.441
        West Point: x = -122.436, y =  47.662, dz =  -1.003
```

In [21]:
```
for s in fault.subfaults:
    s.depth = 5000.
depths = [s.depth for s in fault.subfaults]
for s in fault.subfaults:
    s.calculate_geometry()
print("subfault depths: ", depths)
print("Mw = %5.2f" % fault.Mw())

fault.create_dtopography(x,y,times)
dtopo = fault.dtopo

for pt in pts:
    Xi,Yj,dz = eval_dtopo(pt[1],pt[2], dtopo)
    print('%s: x = %7.3f, y = %7.3f, dz = %7.3f' % (pt[0].rjust(18), Xi,
Yj,dz))
```

```
subfault depths:  [5000.0, 5000.0, 5000.0, 5000.0, 5000.0, 5000.0]
Mw =  7.39
        Alki Point: x = -122.421, y =  47.574, dz =   2.910
 Restoration Point: x = -122.481, y =  47.584, dz =   4.127
        West Point: x = -122.436, y =  47.662, dz =  -0.591
```

Experiment with different rakes:

In [22]:
```
fault = deepcopy(fault_original)
rakes = [s.rake for s in fault.subfaults]
print("subfault rakes: ", rakes)
```

```
subfault rakes:  [90.0, 90.0, 90.0, 90.0, 90.0, 90.0]
```

In [23]:
```
for s in fault.subfaults:
    s.rake = 60.
    s.calculate_geometry()
rakes = [s.rake for s in fault.subfaults]
print("subfault rakes: ", rakes)
print("Mw = %5.2f" % fault.Mw())

fault.create_dtopography(x,y,times)
dtopo = fault.dtopo

for pt in pts:
    Xi,Yj,dz = eval_dtopo(pt[1],pt[2], dtopo)
    print('%s: x = %7.3f, y = %7.3f, dz = %7.3f' % (pt[0].rjust(18), Xi,
Yj,dz))
```

```
subfault rakes:  [60.0, 60.0, 60.0, 60.0, 60.0, 60.0]
Mw =  7.39
        Alki Point: x = -122.421, y =  47.574, dz =   2.980
 Restoration Point: x = -122.481, y =  47.584, dz =   3.740
        West Point: x = -122.436, y =  47.662, dz =  -0.512
```

Scale down slip to obtain Mw 7.3 fault with 35 km width

In [77]:
```
fault = deepcopy(fault_original)
for s in fault.subfaults:
    s.slip *= 0.5
    s.calculate_geometry()

slips = [s.slip for s in fault.subfaults]
print("subfault slips: ", slips)
print("Mw = %5.2f" % fault.Mw())
```

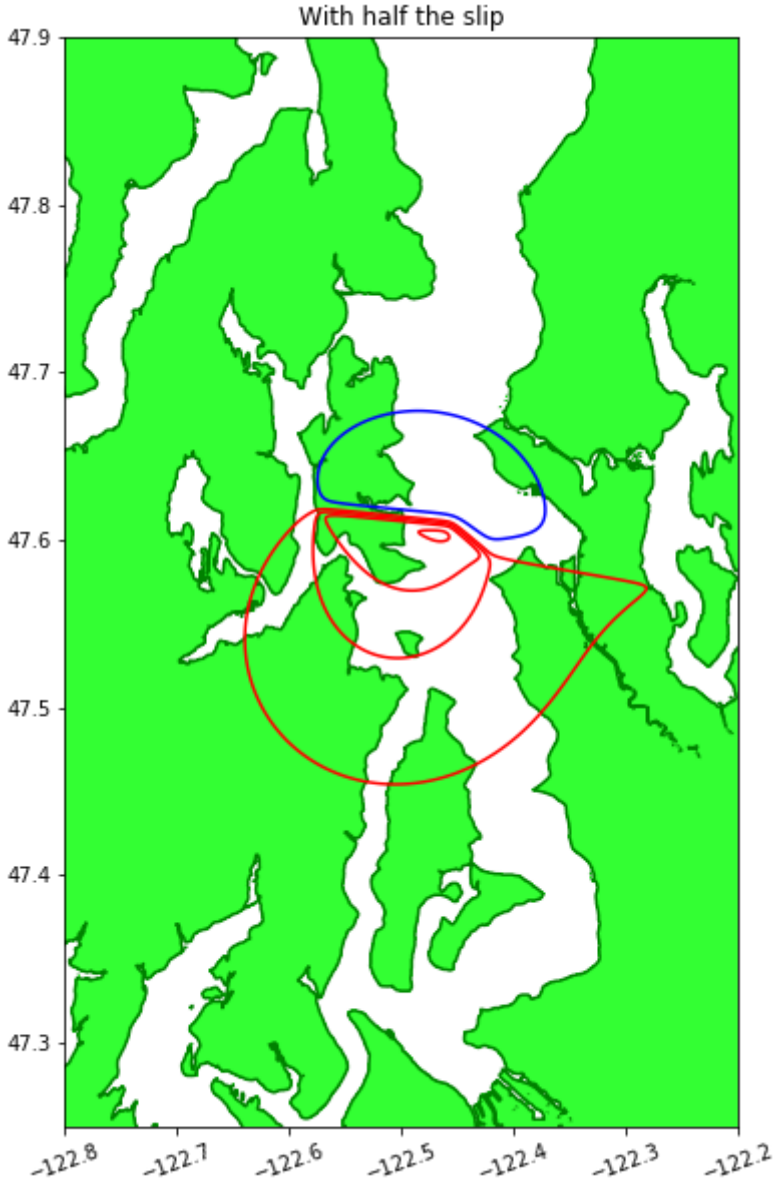
```
subfault slips:  [0.5, 0.5, 6.0, 5.5, 2.0, 0.5]
Mw =  7.35
```

In [78]:
```
fault.create_dtopography(x,y,times)
dtopo = fault.dtopo

for pt in pts:
    Xi,Yj,dz = eval_dtopo(pt[1],pt[2], dtopo)
    print('%s: x = %7.3f, y = %7.3f, dz = %7.3f' % (pt[0].rjust(18), Xi,
Yj,dz))
```

```
        Alki Point: x = -122.421, y =  47.574, dz =   1.878
 Restoration Point: x = -122.481, y =  47.584, dz =   3.464
        West Point: x = -122.436, y =  47.662, dz =  -0.549
```

In [79]:
```
figure(figsize=(15,10))
clf()
subplot(1,2,1)
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo min, max: ',dtopo.dZ.min(),dtopo.dZ.max())

clines = arange(-2,0,0.5)
print('blue clines = ',clines)
CB = contour(dtopo.X, dtopo.Y, dtopo.dZ[-1,:,:], clines,
        colors='b', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)
CR = contour(dtopo.X, dtopo.Y, dtopo.dZ[-1,:,:], clines,
        colors='r', linestyles='-')
#clabel(CR,clines[::3])

ticklabel_format(format='plain',useOffset=False)

#xticks(arange(-122.6,-122.14,.06))
#yticks(arange(47.8,48.35,.04))
xticks(rotation=20,fontsize=10)
yticks(fontsize=10)
gca().set_aspect(1./cos(48*pi/180.))
title('With half the slip')

savefig('SFL-halfslip.png',bbox_inches='tight')
```

```
dtopo min, max:  -0.892246517166 4.07892054369
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
```



Deformation from Tim Walsh:

In [49]:
```
deform = topotools.Topography()
deform.read(\
  '/Users/rjl/B/dtopo/SeattleFault_Mw7.3/TimWalsh/modelresults/A_seafaul
tM73/everett_a_deformation.asc',3)
```

In [50]:
```
print("X limits: %12.2f, %12.2f" % (deform.X.min(), deform.X.max()))
print("Y limits: %12.2f, %12.2f" % (deform.Y.min(), deform.Y.max()))
print("Z limits: %12.2f, %12.2f" % (deform.Z.min(), deform.Z.max()))
```

```
X limits:   1177351.23,   1324631.74
Y limits:    115286.46,    482331.08
Z limits:     -9999.00,         8.35
```

In [51]:
```
import pyproj
from scipy.interpolate import RegularGridInterpolator
```

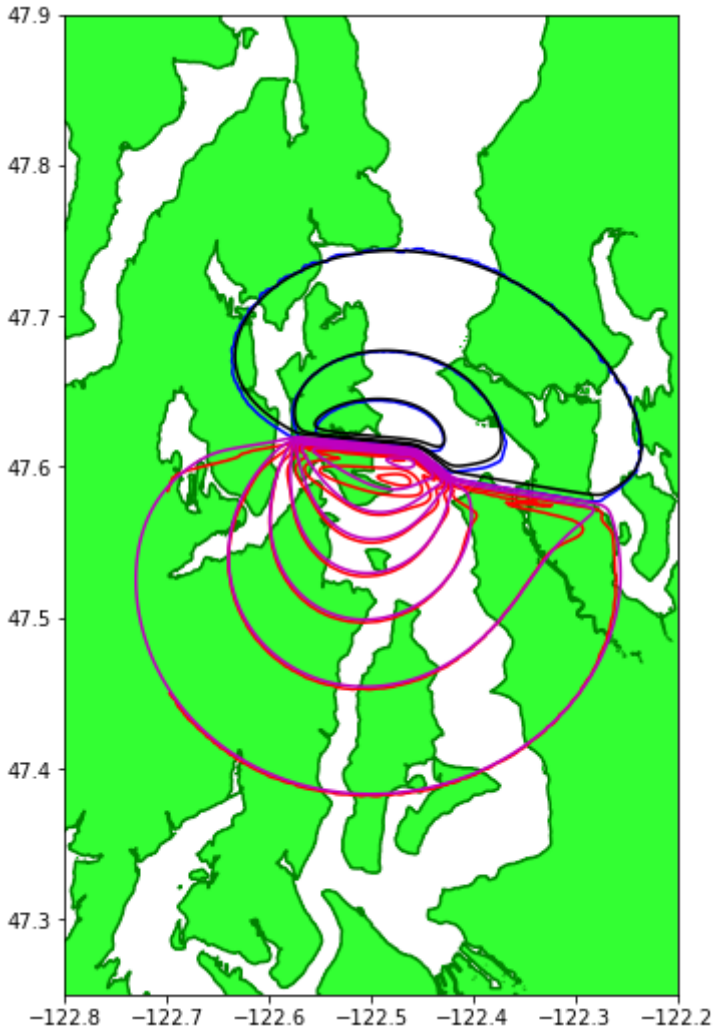
In [52]:
```
# WAN below refers to the Washington State Plane North projection
# See http://spatialreference.org/ref/epsg/2926/
WASHINGTON_NORTH = 'epsg:2926'
WASHINGTON_SOUTH = 'epsg:2286'

#WASHINGTON_NORTH = 'epsg:102748'


proj_WAN = pyproj.Proj(init=WASHINGTON_NORTH)
#proj_WAN = pyproj.Proj(" +proj=lcc +lat_1=47.5 +lat_2=48.73333333333333
 +lat_0=47 +lon_0=-120.8333333333333" \
#           + "+x_0=500000.0000000002 +y_0=0 +ellps=GRS80 +datum=NAD83 +
preserve_units=True no_defs")
```

In [53]:
```
# This file is on a 30-foot grid
# shift x,y points to cell centers
from copy import deepcopy
WANtopo = deepcopy(deform)
WANtopo._x += 15.
WANtopo._y += 15.

# Convert from feet to meters:
WANtopo._x *= 0.3048
WANtopo._y *= 0.3048
WANtopo._delta = None
print('delta in meters = ', WANtopo.delta)

# Convert Z from feet to meters, leaving missing data values alone:
WANtopo._Z = where(WANtopo._Z != WANtopo.no_data_value, WANtopo._Z * 0.3
048, WANtopo.no_data_value)
WANtopo.generate_2d_coordinates()

Z = where(WANtopo.Z > -9998, WANtopo.Z, nan)
WANtopo_func = RegularGridInterpolator((WANtopo.x,WANtopo.y), Z.T,
                                       bounds_error=False,
                                       fill_value=nan)

x = numpy.arange(-123, -122.2, 2./3600.)
y = numpy.arange(47., 48.6, 2./3600.)
X_desired, Y_desired = meshgrid(x,y)

# convert desired lat-long grid to (nonuniform) WA North grid:
x_desired_WAN, y_desired_WAN = proj_WAN(X_desired, Y_desired)

# interpolate to these points:
print('Interpolating points...')
xy_desired_WAN = vstack((ravel(x_desired_WAN), ravel(y_desired_WAN))).T
z_desired_WAN = WANtopo_func(xy_desired_WAN)
Z_desired = reshape(z_desired_WAN, x_desired_WAN.shape)
#Z_desired = where(isnan(Z_desired), -9999, Z_desired)
```

```
delta in meters =  (235.03193414182169, 235.03193414186535)
Interpolating points...
```

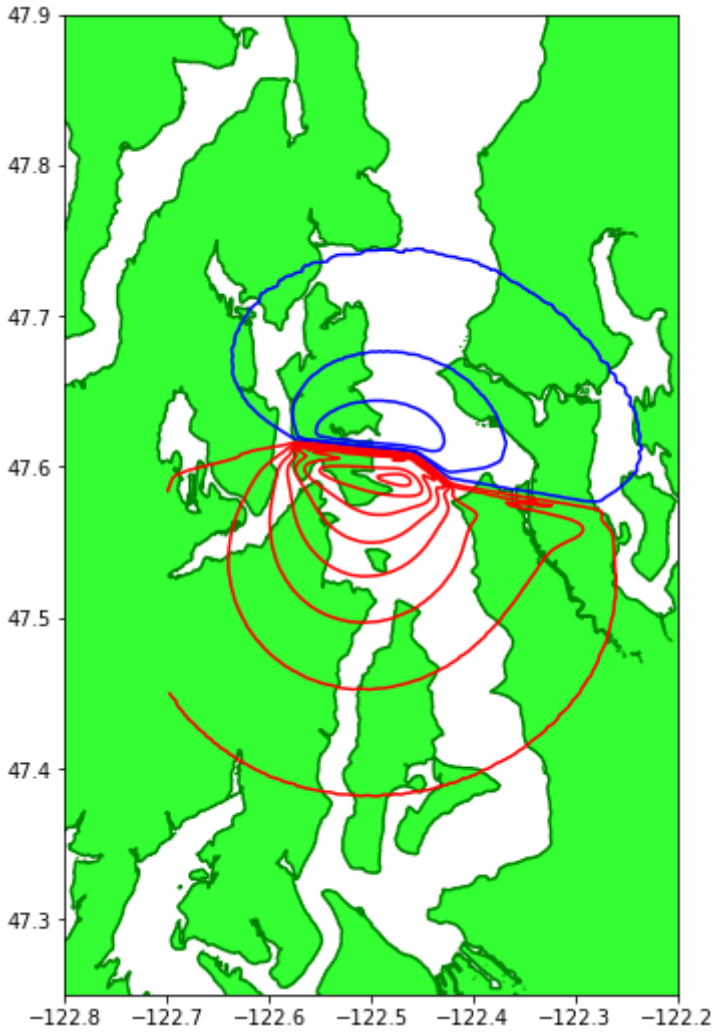
In [66]:
```
figure(figsize=(14,9))
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

#dxc = 0.05
#clines = hstack((arange(-0.3,0.,dxc),arange(dxc,0.3,dxc)))
#contour(X_desired, Y_desired, Z_desired,clines,colors='r')

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo_original min, max: ',dtopo_original.dZ.min(),dtopo_original
.dZ.max())
print('Z_desired min, max: ',nanmin(Z_desired),nanmax(Z_desired))


clines = arange(-2,0,0.5)
print('blue clines = ',clines)
CB = contour(X_desired, Y_desired, Z_desired/.3048, clines,
        colors='b', linestyles='-')
CB = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-1
,:,:], clines,
        colors='k', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)
CR = contour(X_desired, Y_desired, Z_desired/.3048, clines,
        colors='r', linestyles='-')
CB = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-1
,:,:], clines,
        colors='m', linestyles='-')
#clabel(CR,clines[::3])

gca().set_aspect(1./cos(47.5*pi/180))
#xlim(-123.2,-122.16)
#ylim(47,48)
print(nanmin(Z_desired), nanmax(Z_desired))
#savefig('SFL-DNR.png', bbox_inches='tight')
```

```
dtopo_original min, max:  -1.78449303433 8.15784108738
Z_desired min, max:  -0.541695790935 2.54367269384
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
-0.541695790935 2.54367269384
```



In [69]:
```
figure(figsize=(14,9))
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

#dxc = 0.05
#clines = hstack((arange(-0.3,0.,dxc),arange(dxc,0.3,dxc)))
#contour(X_desired, Y_desired, Z_desired,clines,colors='r')

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo_original min, max: ',dtopo_original.dZ.min(),dtopo_original
.dZ.max())
print('Z_desired min, max: ',nanmin(Z_desired),nanmax(Z_desired))


clines = arange(-2,0,0.5)
print('blue clines = ',clines)
CB = contour(X_desired, Y_desired, Z_desired/.3048, clines,
        colors='b', linestyles='-')
#CB = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-
1,:,:], clines,
#       colors='k', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)
CR = contour(X_desired, Y_desired, Z_desired/.3048, clines,
        colors='r', linestyles='-')
#CB = contour(dtopo_original.X, dtopo_original.Y, dtopo_original.dZ[-
1,:,:], clines,
#        colors='m', linestyles='-')
#clabel(CR,clines[::3])

gca().set_aspect(1./cos(47.5*pi/180))
#xlim(-123.2,-122.16)
#ylim(47,48)
print(nanmin(Z_desired), nanmax(Z_desired))
savefig('SFL-DNR.png', bbox_inches='tight')
```

```
dtopo_original min, max:  -1.78449303433 8.15784108738
Z_desired min, max:  -0.541695790935 2.54367269384
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
-0.541695790935 2.54367269384
```



deformation file from PMEL

After resampling on uniform grid. This deformation was used for the Snohomish study.

In [81]:
```
dtopo_unif = dtopotools.DTopography()
dtopo_unif.read('/Users/rjl/git/WA_EMD_Snohomish/dtopo/dtopofiles/seattl
efault_uniform.tt3')
```

In [83]:
```
figure(figsize=(14,9))
for topo in [topo1,topo2,topo3]:
    contourf(topo.X,topo.Y,topo.Z,[0,1e6],colors=[[.2,1,.2]])
    contour(topo.X,topo.Y,topo.Z,[0],colors='g',lw=0.2)

#dxc = 0.05
#clines = hstack((arange(-0.3,0.,dxc),arange(dxc,0.3,dxc)))
#contour(X_desired, Y_desired, Z_desired,clines,colors='r')

xlim(-122.8,-122.2)
ylim(47.25,47.9)
print('dtopo_unif min, max: ',dtopo_unif.dZ.min(),dtopo_unif.dZ.max())
print('Z_desired min, max: ',nanmin(Z_desired),nanmax(Z_desired))


clines = arange(-2,0,0.5)
print('blue clines = ',clines)

CB = contour(dtopo_unif.X, dtopo_unif.Y, dtopo_unif.dZ[-1,:,:], clines,
       colors='b', linestyles='-')
#clabel(CB,clines[::3])

clines = arange(1,9,1)
print('red clines = ',clines)

CB = contour(dtopo_unif.X, dtopo_unif.Y, dtopo_unif.dZ[-1,:,:], clines,
        colors='r', linestyles='-')
#clabel(CR,clines[::3])

gca().set_aspect(1./cos(47.5*pi/180))
#xlim(-123.2,-122.16)
#ylim(47,48)
print(nanmin(Z_desired), nanmax(Z_desired))
savefig('SFL-unif.png', bbox_inches='tight')
```

```
dtopo_unif min, max:  -1.775052 8.372348
Z_desired min, max:  -0.541695790935 2.54367269384
blue clines =  [-2.  -1.5 -1.  -0.5]
red clines =  [1 2 3 4 5 6 7 8]
-0.541695790935 2.54367269384
```