```
######################################################################
# Step 3: Optimize SNF Supply Logistics
# Makes Plots
# Plots for minimum cost while treating all cases
######################################################################

import csv
from math import sqrt
from sys import exit
import pickle
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.mlab as mlab
import os
from scipy.stats import norm
import matplotlib as mpl
import matplotlib.cm as cm
import matplotlib.colors as colors
import shapefile
from math import sin, cos, sqrt, atan2, radians, pi, degrees
from scipy import ndimage
from matplotlib.font_manager import FontProperties
import cartopy.crs as ccrs
import cartopy.io.shapereader as shpreader
import itertools


################################################3
# Functions
################################################3
def make_cmap(colors, position=None, bit=False):
    '''
    make_cmap takes a list of tuples which contain RGB values. The RGB
    values may either be in 8-bit [0 to 255] (in which bit must be set to
    True when called) or arithmetic [0 to 1] (default). make_cmap returns
    a cmap with equally spaced colors.
    Arrange your tuples so that the first color is the lowest value for the
    colorbar and the last is the highest.
    position contains values from 0 to 1 to dictate the location of each color.
    '''
    import matplotlib as mpl
    import numpy as np
    bit_rgb = np.linspace(0,1,256)
    if position == None:
        position = np.linspace(0,1,len(colors))
    else:
        if len(position) != len(colors):
            sys.exit("position length must be the same as colors")
        elif position[0] != 0 or position[-1] != 1:
            sys.exit("position must start with 0 and end with 1")
    if bit:
        for i in range(len(colors)):
            colors[i] = (bit_rgb[colors[i][0]],
                         bit_rgb[colors[i][1]],
                         bit_rgb[colors[i][2]])
    cdict = {'red':[], 'green':[], 'blue':[]}
    for pos, color in zip(position, colors):
        cdict['red'].append((pos, color[0], color[0]))
        cdict['green'].append((pos, color[1], color[1]))
        cdict['blue'].append((pos, color[2], color[2]))

    cmap = mpl.colors.LinearSegmentedColormap('my_colormap',cdict,256)
    return cmap
################################################3
```

```
try:
    wddata='/Users/lilllianpetersen/iiasa/data/supply_chain/'
    wdfigs='/Users/lilllianpetersen/iiasa/figs/supply_chain/'
    wdvars='/Users/lilllianpetersen/iiasa/saved_vars/supply_chain/'
    f=open(wddata+'population/CAPITALVERSIONcasenumbers.csv','r')
except:
    wddata='C:/Users/garyk/Documents/code/riskAssessmentFromPovertyEstimations/supply_chain/data/'
    wdfigs='C:/Users/garyk/Documents/code/riskAssessmentFromPovertyEstimations/supply_chain/figs/'
    wdvars='C:/Users/garyk/Documents/code/riskAssessmentFromPovertyEstimations/supply_chain/vars/'

MakePlots=False
MakeLinePlots=False
MakeStackedBarPlots=False
MakeExportPlots=False
MakeSkeleton=False
MakeByFactoryPlots=False

subsaharancountry = np.load(wdvars+'subsaharancountry.npy')

#for f in range(len(subsaharancountry)):
#    subsaharancountry[f]=subsaharancountry[f].replace(' ','_')
subsaharancountry[subsaharancountry=='Congo']='DRC'
subsaharancountry[subsaharancountry=='Congo (Republic of the)']='Congo'
subsaharancountry[subsaharancountry=="Cote d'Ivoire"]='Ivory Coast'

countrycosted=np.load(wdvars+'countrycosted.npy')
countrycosted[countrycosted=='Congo']='DRC'
countrycosted[countrycosted=='Congo (Republic of the)']='Congo'
countrycosted[countrycosted=="I_Cote d'Ivoire"]='I_Ivory Coast'
countrycosted[countrycosted=="Cote d'Ivoire"]='Ivory Coast'

capitalcosted=np.load(wdvars+'capitalcosted.npy')
subsaharancapital=np.load(wdvars+'subsaharancapital.npy')

try:
    capitalLatLon = np.load(wdvars+'capitalLatLon.npy')
except:
    capitalLatLon=np.zeros(shape=(2,len(capitalcosted)))
    for c in range(len(capitalcosted)):
        if countrycosted[c][:2]=='I_':
            location = geolocator.geocode(capitalcosted[c]+', '+countrycosted[c][2:])
        else:
            location = geolocator.geocode(capitalcosted[c]+', '+countrycosted[c])
        capitalLatLon[0,c] = location.latitude
        capitalLatLon[1,c] = location.longitude
    np.save(wdvars+'capitalLatLon.npy',capitalLatLon)

try:
    SScapitalLatLon = np.load(wdvars+'subsaharancapitalLatLon.npy')
except:
    SScapitalLatLon=np.zeros(shape=(2,len(subsaharancapital)))
    for c in range(len(subsaharancountry)):
        location = geolocator.geocode(subsaharancapital[c]+', '+subsaharancountry[c])
        SScapitalLatLon[0,c] = location.latitude
        SScapitalLatLon[1,c] = location.longitude
    np.save(wdvars+'subsaharancapitalLatLon.npy',SScapitalLatLon)

#  'AllIntl_opti',
optiLevel = ['AllarOpti','LocalOpti', 'AllIntl']
loopvar = ['shipcost','impexp','strtup','truckfactor','tariff']

LTitles = ['All Optimized','Local Optimized','Current Intl']
VTitles = ['Shipping','Import/Export','Startup','Trucking', 'Tariff']

Ltitles = ['AllOpti','LocalOpti', 'AllIntl']
```

```
Vtitles = ['shipping','importexport','startup','trucking', 'tariff']

mins= np.array([0.2,0.2,0.5,0.2, 0.2])
factor = np.array([0.2,0.2,0.5,0.2, 0.2])
maxs = np.array([2.01,4.01,9.51,4.01, 2.6])
#LTitles = ['All Optimized','Local Optimized','Local Producing Optimized International with Tariff','Loca
#optiLevel = ['AllarOpti','LocalOpti', 'AllIntl_opti_trf', 'AllIntl_trf', 'LocalOpti_trf', 'AllarOpti_trf

#[AllOpti,LocalOpti,AllIntl,AllIntlOpti] , [Trf,NoTrf] , [Shipcost,imexp,strtup,truckfactor] , [scenarios
cost=np.zeros(shape=(4,5,20))
costOneAll=np.zeros(shape=(4))
Mask=np.ones(shape=(4,5,20),dtype=bool)
factoryNum=np.zeros(shape=(4,5,20))
factoryNumOneAll=np.zeros(shape=(4))
portNumOneAll=np.zeros(shape=(4))
pctLocal=np.zeros(shape=(4,5,20,2))
pctLocalOneAll=np.zeros(shape=(4,2)) #optiLevel, trf, SAM-MAM

for L in range(len(optiLevel)):
    for V in range(len(loopvar)):
        File =optiLevel[L]+'_'+loopvar[V]
        print optiLevel[L],loopvar[V]

        shp=len(np.arange(mins[V],maxs[V],factor[V]))
        pctTrans=np.zeros(shape=(shp))
        pctIngredient=np.zeros(shape=(shp))
        avgShipments=np.zeros(shape=(shp))
        sizeAvg=np.zeros(shape=(2,shp))
        sizeStdDev=np.zeros(shape=(2,shp))
        sizeMinMax=np.zeros(shape=(2,shp,2))
        factorySizeAll=np.zeros(shape=(2,shp,33))
        factorySizeAllMask=np.ones(shape=(2,shp,33),dtype=bool)

        factoryPct=np.zeros(shape=(2,shp,len(countrycosted)))
        factoryPctOne=np.zeros(shape=(2,len(countrycosted)))
        capacityOne=np.zeros(shape=2)

        for f in range(len(countrycosted)):
            countrycosted[f]=countrycosted[f].replace(' ','_')

        i=-1
        for s in np.arange(mins[V],maxs[V],factor[V]):
            s=np.round(s,2)
            #print '\n',s
            i+=1
            countriesWfactories=[]
            factorySizeS=[]
            factorySizeM=[]
            capacity=np.zeros(shape=2)

            try:
                f = open(wddata+'results2/'+str(File)+'/'+str(File)+str(s)+'.csv')
            except:
                continue
            k=-1
            j=-1
            for line in f:
                k+=1
                tmp=line.split(',')
                if k==0:
                    cost[L,V,i]=float(tmp[1])
                    Mask[L,V,i]=0
                    if s==1:
                        costOne=float(tmp[1])
                        costOneAll[L]=float(tmp[1])
```

```
            #print 'cost',cost[i]
        elif k==1:
            factoryNum[L,V,i]=float(tmp[1])
            if s==1:
                factoryNumOneAll[L]=float(tmp[1])
            #print 'factoryNum',factoryNum[i]
        elif k==2:
            pctTrans[i]=float(tmp[1])
            #print 'pctTrans',pctTrans[i]
        elif k==3:
            pctIngredient[i]=float(tmp[1])
            #print 'pctIngredient',pctIngredient[i]
        elif k==4:
            avgShipments[i]=float(tmp[1])
            #print 'avgShipments',avgShipments[i]
        else:
            j+=1
            country=tmp[0]
            if country=='Congo': country='DRC'
            if country=='Congo_(Republic_of_the)': country='Congo'
            if country=="I_Cote_d'Ivoire": country='I_Ivory_Coast'
            if country=="Cote_d'Ivoire": country='Ivory_Coast'

            c=np.where(country==countrycosted)[0][0]

            #country=country.replace('_',' ')
            countriesWfactories.append(country)
            factorySizeS.append(float(tmp[1]))
            factorySizeM.append(float(tmp[2]))
            factorySizeAll[0,i,c]=float(tmp[1])
            factorySizeAll[1,i,c]=float(tmp[2])
            factorySizeAllMask[:,i,c]=False
            capacity[0]+=float(tmp[1])
            capacity[1]+=float(tmp[2])

            factoryPct[0,i,c]=float(tmp[1])
            factoryPct[1,i,c]=float(tmp[2])
            if s==1:
                factoryPctOne[0,c]=float(tmp[1])
                factoryPctOne[1,c]=float(tmp[2])
                capacityOne[0]+=float(tmp[1])
                capacityOne[1]+=float(tmp[2])
                if country[:2]!='I_':
                    pctLocalOneAll[L,0]+=float(tmp[1])
                    pctLocalOneAll[L,1]+=float(tmp[2])
                if country[:2]=='I_' and V==0:
                    portNumOneAll[L]+=1
            if country[:2]!='I_':
                pctLocal[L,V,i,0]+=float(tmp[1])
                pctLocal[L,V,i,1]+=float(tmp[2])

    pctLocal[L,V,i,:]=pctLocal[L,V,i,:]/capacity[:]

    factorySizeS=np.array(factorySizeS)
    factorySizeM=np.array(factorySizeM)
    sizeAvg[0,i]=np.mean(factorySizeS)
    sizeAvg[1,i]=np.mean(factorySizeM)
    sizeStdDev[0,i]=np.std(factorySizeS)
    sizeStdDev[1,i]=np.std(factorySizeM)
    sizeMinMax[0,i,0]=np.amin(factorySizeS)
    sizeMinMax[1,i,0]=np.amin(factorySizeM)
    sizeMinMax[0,i,1]=np.amax(factorySizeS)
    sizeMinMax[1,i,1]=np.amax(factorySizeM)
factorySizeAll=np.ma.masked_array(factorySizeAll,factorySizeAllMask)
pctLocalOneAll[L,:]=pctLocalOneAll[L,:]/capacityOne[:]
```

```
totalCapacity = np.zeros(shape=(2,len(factorySizeAll[0])))
totalCapacity[0] = np.sum(factorySizeAll[0],axis=1)
totalCapacity[1] = np.sum(factorySizeAll[1],axis=1)
factoryPct = np.swapaxes(factoryPct,1,2)
for p in range(33):
    factoryPct[0,p] = 100*factoryPct[0,p]/totalCapacity[0]
    factoryPct[1,p] = 100*factoryPct[1,p]/totalCapacity[1]

if not os.path.exists(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]):
    os.makedirs(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V])
if not os.path.exists(wdfigs+'cost_optimization/'+Ltitles[L]+'/geographical'):
    os.makedirs(wdfigs+'cost_optimization/'+Ltitles[L]+'/geographical')
if not os.path.exists(wdfigs+'cost_optimization/'+Ltitles[L]+'/exports_by_country/'):
    os.makedirs(wdfigs+'cost_optimization/'+Ltitles[L]+'/exports_by_country/')

x = np.arange(mins[V],maxs[V],factor[V])
x = x*100
if MakeLinePlots:
    fig = plt.figure(figsize=(9, 6))
    #### cost ####
    ydata=np.ma.compressed(np.ma.masked_array(cost[L,V,:],Mask[L,V,:]))
    plt.clf()
    plt.plot(x,ydata,'b*-')
    plt.title(LTitles[L]+': Effect of '+VTitles[V]+' Cost on Total Cost')
    plt.xlabel(VTitles[V]+' Cost, % of Today')
    plt.ylabel('Total Procurement Cost for One Year')
    plt.grid(True)
    plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]+'/'+Ltitles[L]+'__totalCost_

    # #### factoryNum ####
    # ydata=np.ma.compressed(np.ma.masked_array(factoryNum[L,V,:],Mask[L,V,:]))
    # plt.clf()
    # plt.plot(x,ydata,'b*-')
    # plt.title(LTitles[L]+': Effect of '+VTitles[V]+' Number of Factories')
    # plt.xlabel(VTitles[V]+' Cost, % of Today')
    # plt.ylabel('Number of Factories')
    # plt.grid(True)
    # plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]+'/'+Ltitles[L]+'__factoryN
    #
    # #### factorySize ####
    # plt.clf()
    # plt.plot(x,sizeAvg[0],'b*-')
    # plt.plot(x,sizeAvg[0]-sizeStdDev[0],'b*--')
    # plt.plot(x,sizeAvg[0]+sizeStdDev[0],'b*--')
    # plt.plot(x,sizeAvg[1],'g*-')
    # plt.plot(x,sizeAvg[1]-sizeStdDev[1],'g*--')
    # plt.plot(x,sizeAvg[1]+sizeStdDev[1],'g*--')
    # plt.title(LTitles[L]+': Avg Factory Size by '+VTitles[V]+' Cost')
    # plt.xlabel(VTitles[V]+' Cost, % of Today')
    # plt.ylabel('Avg Factory Size')
    # plt.grid(True)
    # plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]+'/'+Ltitles[L]+'factorySiz
    #
    # #### factorySizeAll ####
    # for g in range(2):
    #     plt.clf()
    #     plt.plot(x,factorySizeAll[g],'b*')
    #     plt.title(LTitles[L]+': Factory Size by '+VTitles[V]+' Cost')
    #     plt.xlabel(''+VTitles[V]+' Cost, % of Today')
    #     plt.ylabel('Factory Size')
    #     plt.grid(True)
    #     plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]+'/'+Ltitles[L]+'factor
    #
    # #### PctTrans ####
```

```
    # plt.clf()
    # plt.plot(x,pctTrans*100,'b*-')
    # plt.title(LTitles[L]+': % Cost that is Transport by '+VTitles[V]+' Cost')
    # plt.xlabel(''+VTitles[V]+' Cost, % of Today')
    # plt.ylabel('% of Total Cost that is Transport')
    # plt.grid(True)
    # plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]+'/'+Ltitles[L]+'pctTrans_v
    #
    # #### PctIngredient ####
    # plt.clf()
    # plt.plot(x,pctIngredient*100,'b*-')
    # plt.title(LTitles[L]+': % Cost that is Ingredient by '+VTitles[V]+' Cost')
    # plt.xlabel(VTitles[V]+' Cost, % of Today')
    # plt.ylabel('% of Total Cost that is Ingredient')
    # plt.grid(True)
    # plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]+'/'+Ltitles[L]+'pctIngredi

################################################
# Stacked Bar Plots
################################################
#SMtitles=['SAM','MAM']
#for g in range(2):
factoryCountries = []
plt.clf()
fig = plt.figure(figsize=(18, 7))
ax = plt.subplot(1,2,1)

for t in range(len(countrycosted)):
    country = countrycosted[t]
    c=np.where(country==countrycosted)[0][0]
    if country=='Congo (Republic of the)':
        country='RepOfCongo'
    elif country=='Congo':
        country='DRC'
    country=country.replace(' ','_')

    if np.amax(factoryPct[:,c])!=0:
        vars()[country+'Pct'] = np.sum(factoryPct[:,c,:],axis=0)/2
        factoryCountries.append(country)

countryComparison = np.zeros(shape=(len(factoryCountries)))
for q in range(len(factoryCountries)):
    country = factoryCountries[q]
    countryComparison[q] = vars()[country+'Pct'][5]
sIndex=np.argsort(countryComparison)
factoryCountries=np.array(factoryCountries)
factoryCountries=factoryCountries[sIndex][::-1]

OfactoryCountries = []
Otitles = []
for country in factoryCountries:
    if country[:2]!='I_':
        OfactoryCountries.append(country)
        Otitles.append(country+' Factory')
for country in factoryCountries:
    if country[:2]=='I_':
        OfactoryCountries.append(country)
        countryTitle = 'Intl: '+country[2:]+' Port'
        Otitles.append(countryTitle)

if MakeStackedBarPlots:
    Dcolors = ['firebrick','m','darkorange','crimson','yellow','indianred','goldenrod','mediumpur
    Icolors = ['navy','lawngreen','darkgreen','deepskyblue','darkslategray','mediumseagreen','ligh
    if Vtitles[V]=='shipping' or Vtitles[V]=='tariff':
        width = 7
```

```
            plt.bar(100,102,width=width+4,color='k')
        if Vtitles[V]=='startup':
            width = 22
            plt.bar(100,102,width=width+14,color='k')
        if Vtitles[V]=='importexport':
            width = 10
            plt.bar(100,102,width=width+4,color='k')

        pvars=[]
        inter=-1
        domes=-1
        for l in range(len(OfactoryCountries)):
            country = OfactoryCountries[l]
            if country[:2]=='I_':
                inter+=1
                clr=Icolors[inter]
            else:
                domes+=1
                clr=Dcolors[domes]

            if l==0:
                vars()['p'+str(l)] = ax.bar(x, vars()[country+'Pct'], width, color=clr, )
                bottomStuff = vars()[country+'Pct']
            else:
                vars()['p'+str(l)] = ax.bar(x, vars()[country+'Pct'], width, color=clr, bottom = botto
                bottomStuff+=vars()[country+'Pct']

            pvars.append(vars()['p'+str(l)])


        fontP = FontProperties()
        fontP.set_size('small')

        plt.title('Procurement by '+VTitles[V]+' Parameter',fontsize=18)
        plt.xlabel(''+VTitles[V]+' Cost, % of Today')
        plt.ylabel('% of Total Production')
        plt.ylim([0,102])
        plt.text(100,-4,'Today',horizontalalignment='center')
        ax.legend((pvars[::-1]),(Otitles[::-1]),bbox_to_anchor=(1, 0.98),prop=fontP)
        plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/'+Vtitles[V]+'/FactoryPct_vs_'+Vtitles[V

#######################################################################
# MAPS
#######################################################################

countrycosted=np.load(wdvars+'countrycosted.npy')
countrycosted[countrycosted=='Congo']='DRC'
countrycosted[countrycosted=='Congo (Republic of the)']='Congo'
countrycosted[countrycosted=="I_Cote d'Ivoire"]='I_Ivory Coast'
countrycosted[countrycosted=="Cote d'Ivoire"]='Ivory Coast'

############################
# Export
############################
if MakeExportPlots:
    colors = [(255,255,255),(152, 240, 152), (97, 218, 97), (65, 196, 65), (42, 175, 42), (28, 162, 28
    my_cmap = make_cmap(colors,bit=True)
    shapename = 'admin_0_countries'
    countries_shp = shpreader.natural_earth(resolution='110m',
        category='cultural', name=shapename)

    plt.clf()
    cmapArray=my_cmap(np.arange(256))
    cmin=0
    cmax=np.amax(np.sum(factoryPctOne[:,:],axis=0)) #*0.9
```

```
y1=0
y2=255

fig = plt.figure(figsize=(10, 8))
MinMaxArray=np.ones(shape=(3,2))
subPlot1 = plt.axes([0.61, 0.07, 0.2, 0.8])
MinMaxArray[0,0]=cmin
MinMaxArray[1,0]=cmax/1e9
plt.imshow(MinMaxArray,cmap=my_cmap)
plt.colorbar(label='Number of Packets (Billions)')

ax = plt.axes([0.05,0.05,0.8,0.85],projection=ccrs.PlateCarree())
ax.set_extent([-19, 53, -37, 39], ccrs.PlateCarree())
ax.coastlines()

plt.plot(capitalLatLon[1,8], capitalLatLon[0,8], marker='*', markersize=12, color=[97/255., 218/25
plt.plot(capitalLatLon[1,8], capitalLatLon[0,8], marker='*', markersize=7, color='darkred', label=
plt.plot(capitalLatLon[1,30], capitalLatLon[0,30], marker='o', markersize=12, color=[97/255., 218,
plt.plot(capitalLatLon[1,30], capitalLatLon[0,30], marker='o', markersize=7, color='darkred', labe

factoryNumOne=0
IntlNumOne=0

for country in shpreader.Reader(countries_shp).records():
    cName=country.attributes['NAME_LONG']
    if cName[-6:]=='Ivoire':
        cName="Ivory Coast"
    if cName=='Democratic Republic of the Congo':
        cName='DRC'
    if cName=='Republic of the Congo':
        cName='Congo'
    if cName=='eSwatini':
        cName='Swaziland'
    if cName=='The Gambia':
        cName='Gambia'
    if cName=='Somaliland':
        cName='Somalia'
    if np.amax(cName==subsaharancountry)==0:
        continue
    if np.amax(cName==countrycosted)==0:
        x=0
        y=y1+(y2-y1)/(cmax-cmin)*(x-cmin)
        icmap=min(255,int(round(y,1)))
        icmap=max(0,int(round(icmap,1)))
        ax.add_geometries(country.geometry, ccrs.PlateCarree(), edgecolor='black',
            facecolor=[cmapArray[icmap,0],cmapArray[icmap,1],cmapArray[icmap,2]],label=cName)
    else:
        c=np.where(cName==countrycosted)[0][0]
        x=np.sum(factoryPctOne[:,c])
        y=y1+(y2-y1)/(cmax-cmin)*(x-cmin)
        icmap=min(255,int(round(y,1)))
        icmap=max(0,int(round(icmap,1)))
        ax.add_geometries(country.geometry, ccrs.PlateCarree(), edgecolor='black', facecolor=[cmap

        if x!=0:
            size = 15*(1+x/cmax)
            plt.plot(capitalLatLon[1,c], capitalLatLon[0,c], marker='*', markersize=size, color=[(
            factoryNumOne+=1
        if x==0:
            plt.plot(capitalLatLon[1,c], capitalLatLon[0,c], marker='*', markersize=7, color='dark


for icoast in range(24,len(countrycosted)):
    x=np.sum(factoryPctOne[:,icoast])
    y=y1+(y2-y1)/(cmax-cmin)*(x-cmin)
```

```
            icmap=min(255,int(round(y,1)))
            icmap=max(0,int(round(icmap,1)))
            if x!=0:
                size = 15*(0.8+x/cmax)
                plt.plot(capitalLatLon[1,icoast], capitalLatLon[0,icoast], marker='o', markersize=size, c
                IntlNumOne+=1
            if x==0:
                plt.plot(capitalLatLon[1,icoast], capitalLatLon[0,icoast], marker='o', markersize=7, colo

        local = str(int(np.round(100*np.sum(factoryPctOne[:,:24])/np.sum(factoryPctOne[:,:]),0)))
        intl = str(np.round(100*np.sum(factoryPctOne[:,24:])/np.sum(factoryPctOne[:,:]),0))
        #costOne = str(int(round(costOne/1000000.,0)))

        plt.title('Production of Treatment by Factory and Port', fontsize=18)
        plt.legend(loc = 'lower left')
        #plt.text(-15,-10,str(factoryNumOne)+' Factories Open\n'+str(IntlNumOne)+' Ports Open\n'+local+'%
        plt.text(-15,-10,str(factoryNumOne)+' Factories Open\n'+str(IntlNumOne)+' Ports Open\n'+local+'% I

        plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/geographical/Export_map.pdf')

    ############################
    # Skeleton
    ############################
    if MakeSkeleton:
        shapename = 'admin_0_countries'
        countries_shp = shpreader.natural_earth(resolution='110m',
            category='cultural', name=shapename)

        plt.clf()
        ax = plt.axes([0.05,0.05,0.8,0.85],projection=ccrs.PlateCarree())
        ax.set_extent([-19, 53, -37, 39], ccrs.PlateCarree())
        ax.coastlines()

        plt.plot(capitalLatLon[1,8], capitalLatLon[0,8], marker='*', markersize=9, color='orangered', labe
        plt.plot(capitalLatLon[1,30], capitalLatLon[0,30], marker='o', markersize=8, color='dodgerblue', l
        plt.plot(SScapitalLatLon[1,9],SScapitalLatLon[0,9], marker='^', markersize=8, color='mediumpurple

        for country in shpreader.Reader(countries_shp).records():
            cName=country.attributes['NAME_LONG']
            if cName[-6:]=='Ivoire':
                cName="Ivory Coast"
            if cName=='Democratic Republic of the Congo':
                cName='DRC'
            if cName=='Republic of the Congo':
                cName='Congo'
            if cName=='eSwatini':
                cName='Swaziland'
            if cName=='The Gambia':
                cName='Gambia'
            if cName=='Somaliland':
                cName='Somalia'
            if np.amax(cName==subsaharancountry)==0:
                continue
            if np.amax(cName==countrycosted)!=0:
                c=np.where(cName==countrycosted)[0][0]
                lon1=capitalLatLon[1,c]
                lat1=capitalLatLon[0,c]
                for iSS in range(len(subsaharancountry)):
                    lat2=SScapitalLatLon[0,iSS]
                    lon2=SScapitalLatLon[1,iSS]
                    dist=np.sqrt((lat2-lat1)**2+(lon2-lon1)**2)
                    if dist<15:
                        plt.plot([lon1,lon2] , [lat1,lat2], color='gray', linestyle='--', linewidth = 0.5

        for icoast in range(24,len(countrycosted)):
```

```
        lon1=capitalLatLon[1,icoast]
        lat1=capitalLatLon[0,icoast]
        for iSS in range(len(subsaharancountry)):
            lat2=SScapitalLatLon[0,iSS]
            lon2=SScapitalLatLon[1,iSS]
            dist=np.sqrt((lat2-lat1)**2+(lon2-lon1)**2)
            if dist<17:
                plt.plot([lon1,lon2] , [lat1,lat2], color='gray', linestyle='--', linewidth = 0.5, tra

    for country in shpreader.Reader(countries_shp).records():
        cName=country.attributes['NAME_LONG']
        if cName[-6:]=='Ivoire':
            cName="Ivory Coast"
        if cName=='Democratic Republic of the Congo':
            cName='DRC'
        if cName=='Republic of the Congo':
            cName='Congo'
        if cName=='eSwatini':
            cName='Swaziland'
        if cName=='The Gambia':
            cName='Gambia'
        if cName=='Somaliland':
            cName='Somalia'
        if np.amax(cName==subsaharancountry)==0:
            continue
        if np.amax(cName==countrycosted)==0:
            ax.add_geometries(country.geometry, ccrs.PlateCarree(), edgecolor='black',
                facecolor='lightgray')
            c=np.where(cName==subsaharancountry)[0][0]
            plt.plot(SScapitalLatLon[1,c],SScapitalLatLon[0,c], marker='^', markersize=8, color='mediu
        else:
            c=np.where(cName==countrycosted)[0][0]
            ax.add_geometries(country.geometry, ccrs.PlateCarree(), edgecolor='black', facecolor='ligh
            plt.plot(capitalLatLon[1,c], capitalLatLon[0,c], marker='*', markersize=9, color='orangere

    for icoast in range(24,len(countrycosted)):
        plt.plot(capitalLatLon[1,icoast], capitalLatLon[0,icoast], marker='o', markersize=8, color='d

    plt.title('Supply Chain Optimization\nPossible Factory and Port Locations')
    plt.legend(loc = 'lower left')
    plt.text(-15,-10,'24 Possible Factories\n9 Possible Ports', bbox=dict(fc="none", boxstyle="round")
    plt.savefig(wdfigs+'cost_optimization/'+'skeleton_map.pdf')

##########################
# Supply Zones
##########################
if MakeExportPlots:
    ruftitles=['rutf','rusf']
    for g in range(2):
        vars()['productarray'+ruftitles[g]] = np.load(wddata+'results2/example/'+optiLevel[L]+'/RN'+ru
        Rcountrycosted1=np.load(wddata+'results2/example/'+optiLevel[L]+'/RNcountry.npy')
        Rsubsaharancountry1=np.load(wddata+'results2/example/'+optiLevel[L]+'/RNsubsaharancountry.npy
        Rcountrycosted=[]
        for i in range(len(Rcountrycosted1)):
            country=Rcountrycosted1[i]
            if country[:2]=='I_':
                countrytmp=country[2:].replace('_',' ')
                Rcountrycosted.append('I_'+countrytmp)
            else:
                countrytmp=country.replace('_',' ')
                Rcountrycosted.append(countrytmp)
        Rcountrycosted=np.array(Rcountrycosted)
        Rsubsaharancountry=[]
        for i in range(len(Rsubsaharancountry1)):
            country=Rsubsaharancountry1[i]
```

```
        if country[:2]=='I_':
            countrytmp=country[2:].replace('_',' ')
            Rsubsaharancountry.append('I_'+countrytmp)
        else:
            countrytmp=country.replace('_',' ')
            Rsubsaharancountry.append(countrytmp)
    Rsubsaharancountry=np.array(Rsubsaharancountry)

    Rsubsaharancountry[Rsubsaharancountry=='Congo']='DRC'
    Rsubsaharancountry[Rsubsaharancountry=='Congo (Republic of the)']='Congo'
    Rsubsaharancountry[Rsubsaharancountry=="Cote d'Ivoire"]='Ivory Coast'
    Rcountrycosted[Rcountrycosted=='Congo']='DRC'
    Rcountrycosted[Rcountrycosted=='Congo (Republic of the)']='Congo'
    Rcountrycosted[Rcountrycosted=="I_Cote d'Ivoire"]='I_Ivory Coast'
    Rcountrycosted[Rcountrycosted=="Cote d'Ivoire"]='Ivory Coast'

productarray=np.mean([productarrayrutf,productarrayrusf],axis=0)

shapename = 'admin_0_countries'
countries_shp = shpreader.natural_earth(resolution='110m', category='cultural', name=shapename)
colors = [(240,59,32),(252,146,114),(254,178,76),(255,237,160),(35,132,67),(127,255,0),(229,245,2
        # colors = [(240,59,32),(252,146,114),(254,178,76),(255,237,160),(49,163,84),(161,217,155,
# colors = [(128,0,0),(170,110,40),(128,128,0),(0,128,128),(0,0,128),(0,0,128),(0,0,0),(230,25,75,
colors=colors[:len(Rcountrycosted)+1]
my_cmap = make_cmap(colors,bit=True)

plt.clf()
cmapArray=my_cmap(np.arange(256))
cmin=0
cmax=len(Rcountrycosted)
y1=0
y2=255

fig = plt.figure(figsize=(10, 8))
#MinMaxArray=np.ones(shape=(3,2))
#subPlot1 = plt.axes([0.61, 0.07, 0.2, 0.8])
#MinMaxArray[0,0]=cmin
#MinMaxArray[1,0]=cmax
#plt.imshow(MinMaxArray,cmap=my_cmap)
#plt.colorbar()

ax = plt.axes([0.05,0.05,0.8,0.85],projection=ccrs.PlateCarree())
ax.set_extent([-19, 53, -37, 39], ccrs.PlateCarree())
ax.coastlines()

factoryNumOne=0
IntlNumOne=0

for country in shpreader.Reader(countries_shp).records():
    cName=country.attributes['NAME_LONG']
    if cName[-6:]=='Ivoire':
        cName="Ivory Coast"
    if cName=='Democratic Republic of the Congo':
        cName='DRC'
    if cName=='Republic of the Congo':
        cName='Congo'
    if cName=='eSwatini':
        cName='Swaziland'
    if cName=='The Gambia':
        cName='Gambia'
    if cName=='Somaliland':
        cName='Somalia'
    if np.amax(cName==Rsubsaharancountry)==0:
        continue
    else:
```

```
            poz=np.where(cName==Rsubsaharancountry)[0][0]
            c=np.where(productarray[:,poz]==np.amax(productarray[:,poz]))[0][0]
            y=y1+(y2-y1)/(cmax-cmin)*(c-cmin)
            icmap=min(255,int(round(y,1)))
            icmap=max(0,int(round(icmap,1)))
            ax.add_geometries(country.geometry, ccrs.PlateCarree(), edgecolor='black', facecolor=[cmap

    for country in shpreader.Reader(countries_shp).records():
        cName=country.attributes['NAME_LONG']
        if cName[-6:]=='Ivoire':
            cName="Ivory Coast"
        if cName=='Democratic Republic of the Congo':
            cName='DRC'
        if cName=='Republic of the Congo':
            cName='Congo'
        if cName=='eSwatini':
            cName='Swaziland'
        if cName=='The Gambia':
            cName='Gambia'
        if cName=='Somaliland':
            cName='Somalia'
        if np.amax(cName==Rsubsaharancountry)==0:
            continue
        else:
            poz=np.where(cName==Rsubsaharancountry)[0][0]
            c=np.where(productarray[:,poz]==np.amax(productarray[:,poz]))[0][0]
            width=0.3+((1.2*productarray[c,poz])/np.amax(productarray))

            p = np.where(cName==subsaharancountry)[0][0]
            lat2=SScapitalLatLon[0,p]
            lon2=SScapitalLatLon[1,p]

            supplier=Rcountrycosted[c]
            p2=np.where(supplier==countrycosted)[0][0]
            lat1=capitalLatLon[0,p2]
            lon1=capitalLatLon[1,p2]

            dlat=lat2-lat1
            dlon=lon2-lon1
            if dlat!=0:
                plt.arrow(lon1, lat1, dlon, dlat, facecolor='w', edgecolor='k', linestyle='-', width=w

    # Plot Ports
    for f in range(len(Rcountrycosted)):
        factory=Rcountrycosted[f]
        if factory[:2]!='I_':
            continue

        y=y1+(y2-y1)/(cmax-cmin)*(f-cmin)
        icmap=min(255,int(round(y,1)))
        icmap=max(0,int(round(icmap,1)))

        p=np.where(factory==countrycosted)[0][0]

        if np.sum(factoryPctOne[:,p])>0:
            size = 15*(0.8+np.sum(factoryPctOne[:,p])/np.amax(np.sum(factoryPctOne[:,:],axis=0)))
            plt.plot(capitalLatLon[1,p], capitalLatLon[0,p], marker='o', markersize=size, markerfacec
            IntlNumOne+=1

    # Plot Factories
    for f in range(len(Rcountrycosted)):
        factory=Rcountrycosted[f]
        if factory[:2]=='I_':
            continue
        y=y1+(y2-y1)/(cmax-cmin)*(f-cmin)
```

```
            icmap=min(255,int(round(y,1)))
            icmap=max(0,int(round(icmap,1)))

            p=np.where(factory==countrycosted)[0][0]

            if np.sum(factoryPctOne[:,p])>0:
                size = 15*(0.8+np.sum(factoryPctOne[:,p])/np.amax(np.sum(factoryPctOne[:,:],axis=0)))
                plt.plot(capitalLatLon[1,p], capitalLatLon[0,p], marker='*', markersize=size, markerfacec
                factoryNumOne+=1
            #if x==0:
            #    plt.plot(capitalLatLon[1,p], capitalLatLon[0,p], marker='*', markersize=7, color='darkre


        local = str(int(np.round(100*np.sum(factoryPctOne[:,:24])/np.sum(factoryPctOne[:,:]),0)))
        intl = str(np.round(100*np.sum(factoryPctOne[:,24:])/np.sum(factoryPctOne[:,:]),0))
        # costOne = str(int(round(costOne/1000000.,0)))

        plt.legend(bbox_to_anchor=(0.98, 0.8),ncol=1,numpoints=1)

        plt.title('Primary Supplier of Treatment by Country',fontsize=18)
        plt.text(-15,-10,str(factoryNumOne)+' Factories Open\n'+str(IntlNumOne)+' Ports Open\n'+local+'% F
        plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/geographical/Supplyzone_map.pdf')

    ############################
    # By factory import/export
    ############################
    if MakeByFactoryPlots:
        for g in range(2):
            productarray = np.load(wddata+'results2/example/'+optiLevel[L]+'/RN'+ruftitles[g]+'array.npy')
            Rcountrycosted1=np.load(wddata+'results2/example/'+optiLevel[L]+'/RNcountry.npy')
            Rsubsaharancountry1=np.load(wddata+'results2/example/'+optiLevel[L]+'/Rsubsaharancountry.npy')
            Rcountrycosted=[]
            for i in range(len(Rcountrycosted1)):
                country=Rcountrycosted1[i]
                if country[:2]=='I_':
                    countrytmp=country[2:].replace('_',' ')
                    Rcountrycosted.append('I_'+countrytmp)
                else:
                    countrytmp=country.replace('_',' ')
                    Rcountrycosted.append(countrytmp)
            Rcountrycosted=np.array(Rcountrycosted)
            Rsubsaharancountry=[]
            for i in range(len(Rsubsaharancountry1)):
                country=Rsubsaharancountry1[i]
                if country[:2]=='I_':
                    countrytmp=country[2:].replace('_',' ')
                    Rsubsaharancountry.append('I_'+countrytmp)
                else:
                    countrytmp=country.replace('_',' ')
                    Rsubsaharancountry.append(countrytmp)
            Rsubsaharancountry=np.array(Rsubsaharancountry)

            Rsubsaharancountry[Rsubsaharancountry=='Congo']='DRC'
            Rsubsaharancountry[Rsubsaharancountry=='Congo (Republic of the)']='Congo'
            Rsubsaharancountry[Rsubsaharancountry=="Cote d'Ivoire"]='Ivory Coast'
            Rcountrycosted[Rcountrycosted=='Congo']='DRC'
            Rcountrycosted[Rcountrycosted=='Congo (Republic of the)']='Congo'
            Rcountrycosted[Rcountrycosted=="I_Cote d'Ivoire"]='I_Ivory Coast'
            Rcountrycosted[Rcountrycosted=="Cote d'Ivoire"]='Ivory Coast'

            colors = [(255,255,255), (203,208,255), (160,169,255), (121,133,255), (79, 95, 255), (43, 62,
            my_cmap = make_cmap(colors,bit=True)
            shapename = 'admin_0_countries'
            countries_shp = shpreader.natural_earth(resolution='110m',
                category='cultural', name=shapename)
```

```
for f in range(len(productarray)):

    factory = Rcountrycosted[f]

    plt.clf()
    cmapArray=my_cmap(np.arange(256))
    cmin=0
    cmax=np.amax(productarray[f,:]) #*0.9
    if cmax==0:
        continue
    y1=0
    y2=255

    fig = plt.figure(figsize=(10, 8))
    MinMaxArray=np.ones(shape=(3,2))
    subPlot1 = plt.axes([0.61, 0.07, 0.2, 0.8])
    MinMaxArray[0,0]=cmin
    MinMaxArray[1,0]=cmax
    plt.imshow(MinMaxArray,cmap=my_cmap)
    plt.colorbar()

    ax = plt.axes([0.05,0.05,0.8,0.85],projection=ccrs.PlateCarree())
    ax.set_extent([-19, 53, -37, 39], ccrs.PlateCarree())
    ax.coastlines()

    plt.plot(-16.1, -34.7, marker='*', markersize=9, color='limegreen', label='Factory',lines
    plt.plot(-16.1, -34.7, marker='o', markersize=8, color='limegreen', label = 'Intl Shipmen
    plt.plot(-16.1, -34.7, marker='^', markersize=8, color='mediumpurple', label = 'Recieves
    impCountries=[]
    impPct=[]

    for country in shpreader.Reader(countries_shp).records():
        cName=country.attributes['NAME_LONG']
        if cName[-6:]=='Ivoire':
            cName="Ivory Coast"
        if cName=='Democratic Republic of the Congo':
            cName='DRC'
        if cName=='Republic of the Congo':
            cName='Congo'
        if cName=='eSwatini':
            cName='Swaziland'
        if cName=='The Gambia':
            cName='Gambia'
        if np.amax(cName==subsaharancountry)==0:
            continue
        impc=np.where(cName==subsaharancountry)[0][0]
        x=productarray[f,impc]
        y=y1+(y2-y1)/(cmax-cmin)*(x-cmin)
        icmap=min(255,int(round(y,1)))
        icmap=max(0,int(round(icmap,1)))
        ax.add_geometries(country.geometry, ccrs.PlateCarree(), edgecolor='black', facecolor=

        if x!=0:
            impCountries.append(cName)
            impPct.append(x)
            size = 10*(1+x/cmax)
            plt.plot(SScapitalLatLon[1,impc], SScapitalLatLon[0,impc], marker='^', markersize=
            facc=np.where(factory==countrycosted)[0][0]
            if factory[:2]=='I_':
                plt.plot(capitalLatLon[1,facc], capitalLatLon[0,facc], marker='o', markersize=
            else:
                plt.plot(capitalLatLon[1,facc], capitalLatLon[0,facc], marker='*', markersize=

            factoryNumOne+=1
```

```
                #for icoast in range(24,len(countrycosted)):
                #    x=factoryPctOne[g,icoast]
                #    if x!=0:
                #        size = 10*(1+factoryPctOne[g,icoast]/cmax)
                #        plt.plot(capitalLatLon[1,icoast], capitalLatLon[0,icoast], marker='o', markersiz
                #        IntlNumOne+=1
                #    if x==0:
                #        plt.plot(capitalLatLon[1,icoast], capitalLatLon[0,icoast], marker='o', markersiz

                totalshipments = np.sum(productarray[f])
                impPct=np.array(impPct)
                impPct=100*impPct/totalshipments
                order=np.argsort(impPct)
                impPct=impPct[order][::-1]
                impCountries=np.array(impCountries)[order][::-1]
                totalshipments = str(int(round(np.sum(productarray[f])/1000000.)))
                #local = str(int(np.round(100*np.sum(factoryPctOne[g,:24])/np.sum(factoryPctOne[g,:]),0)),
                #intl = str(np.round(100*np.sum(factoryPctOne[g,24:])/np.sum(factoryPctOne[g,:]),0))
                #costOne = str(int(round(costOne/1000000.,0)))

                plt.title('Exports of RUTF for '+factory+', Packets \n' + LTitles[L])
                if factory[:2]=='I_':
                    plt.title(SMtitles[g]+' Treatment Supplied by '+factory[2:]+' Port\n' + LTitles[L])
                    plt.text(-15,-8,factory[2:]+' Port\n'+totalshipments+' Million Packets Procured', size
                    for r in range(len(impCountries)):
                        plt.text(-13,-10.4-1.7*r,'- '+impCountries[r]+', '+str(int(round(impPct[r])))+'%'
                else:
                    plt.title(SMtitles[g]+' Treatment Supplied by '+factory+' Factory\n' + LTitles[L])
                    plt.text(-15,-8,factory+' Factory\n'+totalshipments+' Million Packets Procured', size
                    for r in range(len(impCountries)):
                        plt.text(-13,-10.4-1.7*r,'- '+impCountries[r]+', '+str(int(round(impPct[r])))+'%'

                plt.legend(loc = 'lower left')
                plt.savefig(wdfigs+'cost_optimization/'+Ltitles[L]+'/exports_by_country/'+Ltitles[L]+'_'+

## cost barchart ##
fig = plt.figure(figsize=(6, 5))
plt.clf()
x=np.array([1,2,3])
ydata = (np.array([costOneAll[0],costOneAll[1],costOneAll[2]])/1e9)[::-1]
colors=['g','b','r'][::-1]
plt.bar(x,ydata,color=colors,tick_label=['Current','Local Optimized','All Optimized'])
plt.ylabel('Total Cost of Procurement for 1 Year (Billion USD)')
plt.title('Total Modeled Cost',fontsize=18)
plt.grid(True,linestyle=':')
plt.savefig(wdfigs+'cost_optimization/summary/barchart_cost.pdf')

## % local barchart ##
fig = plt.figure(figsize=(6, 5))
plt.clf()
x=np.array([1,2,3])
pctLocalOneAll1 = np.mean(pctLocalOneAll[:,:],axis=1)
ydata = (np.array([pctLocalOneAll1[0],pctLocalOneAll1[1],pctLocalOneAll1[2]])*100)[::-1]
colors=['g','b','r'][::-1]
plt.bar(x,ydata,color=colors,tick_label=['Current','Local Optimized','All Optimized'])
plt.ylabel('% Treatment Produced Locally')
plt.title('Percent Produced Locally',fontsize=18)
plt.grid(True,linestyle=':')
plt.savefig(wdfigs+'cost_optimization/summary/barchart_pctLocal.pdf')

## factoryNum barchart ##
fig = plt.figure(figsize=(6, 5))
plt.clf()
```

```
x1=np.array([0.79,1.79,2.79])
x2=np.array([1.21,2.21,3.21])

bar_width=0.4
ydata = (np.array([factoryNumOneAll[0],factoryNumOneAll[1],factoryNumOneAll[2]]))[::-1]
colors=['g','b','r'][::-1]
plt.bar(x1,ydata,color=colors,width=bar_width)

ydata = (np.array([portNumOneAll[0],portNumOneAll[1],portNumOneAll[2]]))[::-1]
plt.bar(x2,ydata,color=colors,width=bar_width)

plt.yticks([0,2,4,6,8,10,12,14,16,18,20])
plt.xticks([1,2,3],['Current','Local Optimized','All Optimized'])
plt.text(0.62,0.4,'Factories',size=8)
plt.text(1.62,0.4,'Factories',size=8)
plt.text(2.62,0.4,'Factories',size=8)
plt.text(1.12,0.4,'Ports',size=8)
plt.text(2.12,0.4,'Ports',size=8)
plt.text(3.12,0.4,'Ports',size=8)
plt.ylabel('Number of Factories or Ports')
plt.title('Number of Factories and Ports',fontsize=18)
plt.grid(True,linestyle=':')
plt.savefig(wdfigs+'cost_optimization/summary/barchart_factoryNum.pdf')



fig = plt.figure(figsize=(7, 3.75))
LTitles = ['All Optimized','Local Optimized','Optimized Intl','Current']

cost1=cost/1e9
for V in range(len(loopvar)):
    x = np.arange(mins[V],maxs[V],factor[V])
    x = x*100
    plt.clf()
    plt.plot([100,100],[0.5,1.35],'k-')
    plt.plot(x,np.ma.compressed(np.ma.masked_array(cost1[2,V,:],Mask[2,V,:])),'r*-',label=LTitles[3])
    plt.plot(x,np.ma.compressed(np.ma.masked_array(cost1[1,V,:],Mask[1,V,:])),'b*-',label=LTitles[1])
    plt.plot(x,np.ma.compressed(np.ma.masked_array(cost1[0,V,:],Mask[0,V,:])),'g*-',label=LTitles[0])

    plt.title('Effect of '+VTitles[V]+' on Total Cost',fontsize=15)
    plt.xlabel(VTitles[V]+' Cost, % of Today')
    plt.ylabel('Procurement Cost for One Year (Billion USD)')
    plt.ylim([0.5,1.35])
    plt.grid(True)
    plt.legend(loc='lower right')
    plt.savefig(wdfigs+'cost_optimization/summary/line_totalCost_vs_'+Vtitles[V]+'.pdf')
exit()

for V in range(len(loopvar)):
    x = np.arange(mins[V],maxs[V],factor[V])
    x = x*100
    plt.clf()
    plt.plot(x,np.ma.compressed(np.ma.masked_array(factoryNum[0,V,:],Mask[0,V,:])),'g*-',label=LTitles[0]
    plt.plot(x,np.ma.compressed(np.ma.masked_array(factoryNum[1,V,:],Mask[1,V,:])),'c*-',label=LTitles[1]
    #plt.plot(x,np.ma.compressed(np.ma.masked_array(factoryNum[2,V,:],Mask[2,V,:])),'b*-',label=LTitles[2
    plt.plot(x,np.ma.compressed(np.ma.masked_array(factoryNum[2,V,:],Mask[2,V,:])),'r*-',label=LTitles[3]

    plt.title('Effect of '+VTitles[V]+' on Number of Factories')
    plt.xlabel(VTitles[V]+' Cost, % of Today')
    plt.ylabel('Number of Factories')
    #plt.ylim([0,2e9])
    plt.grid(True)
    plt.legend()
    plt.savefig(wdfigs+'cost_optimization/summary/line_factoryNum_vs_'+Vtitles[V]+'.pdf')
```

```
for V in range(len(loopvar)):
    x = np.arange(mins[V],maxs[V],factor[V])
    x = x*100
    plt.clf()
    plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[0,V,:,0],Mask[0,V,:])),'g*-',label=LTitles
    plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[1,V,:,0],Mask[1,V,:])),'c*-',label=LTitles
    #plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[2,V,:,0],Mask[2,V,:])),'b*-',label=LTitl
    try:
        plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[2,V,:,0],Mask[2,V,:])),'r*-',label=LT:
    except:
        print V
    plt.title('Effect of '+VTitles[V]+' on % RUTF Produced Locally')
    plt.xlabel(VTitles[V]+' Cost, % of Today')
    plt.ylabel('Percent of RUTF Produced Locally')
    plt.ylim([0,101])
    plt.grid(True)
    plt.legend()
    plt.savefig(wdfigs+'cost_optimization/summary/line_0pctLocal_vs_'+Vtitles[V]+'.pdf')

for V in range(len(loopvar)):
    x = np.arange(mins[V],maxs[V],factor[V])
    x = x*100
    plt.clf()
    plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[0,V,:,1],Mask[0,V,:])),'g*-',label=LTitles
    plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[1,V,:,1],Mask[1,V,:])),'c*-',label=LTitles
    #plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[2,V,:,1],Mask[2,V,:])),'b*-',label=LTitl
    try:
        plt.plot(x,100*np.ma.compressed(np.ma.masked_array(pctLocal[2,V,:,1],Mask[2,V,:])),'r*-',label=LT:
    except:
        print V
    plt.title('Effect of '+VTitles[V]+' on % RUSF Produced Locally')
    plt.xlabel(VTitles[V]+' Cost, % of Today')
    plt.ylabel('Percent of RUSF Produced Locally')
    plt.ylim([0,101])
    plt.grid(True)
    plt.legend()
    plt.savefig(wdfigs+'cost_optimization/summary/line_1pctLocal_vs_'+Vtitles[V]+'.pdf')
```