**S3 Appendix: Code**

**List of .m files needed to run ABM**

(1) Hazard\_SEES\_ABM.m

(2) Return\_Damage\_Initial.m

(3) findDamageGivenWS.m

(4) Mitigation\_Scenario2\_Upgrade.m

(5) Mitigation\_Scenario3\_Neighbor\_Mod.m

(6) Mitigation\_Scenario4\_Policy.m

(7) Returns\_NewResistanceLevel.m

(8) CountNumUpgrades\_inReplication.m

(9) save\_data.m

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(1) Hazard\_SEES\_ABM.m**

function Hazard\_SEES\_ABM(caseNo, city, StormWindSpeed, percentHouseGivenSubsidy, ...

Upgrade\_Sensitivity\_factor, threshold\_Neigh, numReplications, ProbMit, FragilityCurve,... Neigh\_List, StormsPerYear, numHurr, Parcel\_Input, numYear, ThresholdElevateMech, ... ThresholdElevateHouse, MinMechProb, MinRaiseProb, parallel, prior, Cost\_route, ...

damage\_prob\_new, Upgrade\_route)

%\*\*\*\*Program Description\*\*\*\*\*\*\*

%this is the master file that, for each replication and year, assesses

%damage (wind) for each parcel, determines which mitigation will

%happen, and then saves the results

%\*\*\*\*\*\*\*\*Inputs\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%numYear = 162 %size: 1 x 1; may be unnecessary if using complete historical record

%numReplications = the number of replications.

%Parcel\_Input: Information on the parcel characteristics. Col1: Parcel ID; Col2:

%    Superseeded? NaN or No which points to new Parcel ID(s);

% Col3: Building Type (1-17)

% Col4: Resitance Level (1 - lowest through 7 - highest);

%    Col5: List of Years Damaged

%StormWindSpeed peak 3s windgust of storm j for household i,

%     size: num households + 1 x num storms. Row 1 represents parcel ID.  Load this

%     if using historical record.

%StormsPerYear    % col1: year;  col2: # storms; col3: 0 if no storms caused damage, ID #

%     of first storm in year i otherwise; col4: 0 if no storms caused damage,

%     ID# of last storm in year i otherwise is now year\_final

%ProbMit: A probability matrix of upgrade likelihood to a higher

%     resistance level after a strom and in DS #: size: no resistance level

%     (to upgrade to) x no damage states x no resistance levels (before

%     storm) USE 'Mit\_Prob' (4 x 1 cell)

%FragilityCurve: the fragility curves, size: 41 x 17 x 7 x 4 (check), 'WFC'

%     in Laiyin's code

%CaseNo: 1, 2, 3, or 4. Size = 1 x 1. if =2, then upgrade case, if =3 then

%   upgrade + neighbor case, if =4 then tax subsidy case

%City: 'AA','Miami'

%Upgrade\_Sensitivity\_factor: the amount to adjust upgrade probabilities by

%   for sensitivity analysis. e.g., 0.1, -0.25 --> 10% inc, 25% dec

%threshold\_Neigh: The threshold of the fraction of neighbors (houses within 75m)

%    who are damaged (at Damage State 2 or above) that above which a parcel will

%    upgrade one resistance level

%Neigh\_List: The list of neighbors (within 75m) for each parcel.

%   Size: num parcels x max(number of neighbors)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%     Parallelization

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if parallel

    %If not inside a PBS job, use 2 processors

    if isempty(getenv('PBS\_NP'))

        NP = 2;

    else

        NP = str2double(getenv('PBS\_NP'));

    end

    myPool = parpool('current',NP);

end

%total number of years with at least one storm

numYearWStorm = sum(StormsPerYear(:,3)>0); %36 initially for Miami case

numStorm = max(StormsPerYear(:,4)); %47 initially for Miami case

numParcelsInitial = size(Parcel\_Input,1);

hurricane\_count = zeros(numParcelsInitial,7);

parfor replication = 1:numReplications  %No. replications

    StormsPerYear\_Loop=StormsPerYear;

    StormWindSpeed\_Loop = StormWindSpeed;

    fprintf(1,'Replication: %d \n',replication);

    Parcel = Parcel\_Input; %Initialize Parcel data

    %initialize building stock for replication

    Parcel = Initialize\_RLofBuildingStock(Parcel);

    Parcel\_forUpgadeCount = Parcel(:,[1 2 3 4]);

    Damage\_storm = zeros(numParcelsInitial,12,numStorm); %records parcel damage in

        %storm x in year 'year.' There could be multiple storms in year 'year'

    Damage\_year = zeros(numParcelsInitial,12,numYearWStorm); %records parcel damage in

        %year 'year.' If there are multiple storms in year 'year' that damage the parcel, the worst

        %damage state is used.

    Upgrade\_year\_tracker = zeros(numParcelsInitial,11);

    Upgrade\_year\_tracker(:,1) = Parcel(:,1);

    Upgrade\_year\_tracker(:,2) = 1;

    Upgrade\_year\_tracker(:,3) = Parcel(:,4);

    yearNext = 0;

    for year = 1:numYear   %No. years of historical hurricane history

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %

        %           Damage

        %

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %Determine how many storms in year j cause damage

        %Use lookup table for no. storms, windspeeds

        %Make a list the storms in 'year' that cause damage

        if StormsPerYear\_Loop(year,3) > 0

            yearNext = yearNext+1; %records num years with storm; max  should be 36

            StormIDsInYear = StormsPerYear\_Loop(year,3):1:StormsPerYear\_Loop(year,4); %List of stormIDs

            %that occur in 'year'

        else

            %no storm occured in 'year' that brought damage

            StormIDsInYear=NaN;

        end

        if ~isnan(StormIDsInYear) %if there is at least one storm

            %Loop determines damage to each house from storm k

            for storm = 1:length(StormIDsInYear)  %for each storm that produces damage in year j

                Damage\_storm(:,:,StormIDsInYear(storm)) = Return\_Damage\_Initial(Parcel,year);

                %WIND DAMAGE

                Damage\_Internal = findDamageGivenWS(Parcel, FragilityCurve, ...

                    StormWindSpeed\_Loop([1;StormIDsInYear(storm)+1],:), year);

                Damage\_storm(:,7:8,StormIDsInYear(storm)) = Damage\_Internal;

                %%%%%%%%%%%%%%%%%%%%%%%%%%%%

            end

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %           2+ Storms in a year

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

            %POPULATE Damage\_year: If House is damaged 2+ times in

            %   'year', record the worse of the damage

            if length(StormIDsInYear) > 1 % if more than 1 storm

                %for each parcel in Damage\_storm in indices 'StormIDsInYear', find the

                %     max damage state (col 7) and the corresponding WS (col 8 and 9)

Damage\_temp = [Damage\_storm(:,1:6,StormIDsInYear(1)) ... zeros(size(Damage\_storm(:,1),1),6)];

                for index = 1:size(Damage\_storm,1) %for each parcel

                    [maxDS, indexMaxDS] = max(Damage\_storm(index,7,StormIDsInYear)); %find max damage

                    %and from which storm

 Damage\_temp(index,7:8) = [maxDS ... Damage\_storm(index,8,StormIDsInYear(indexMaxDS(1)))];

                end

                Damage\_year(:,:,yearNext)=Damage\_temp;

            else %There is only one storm in year 'year'; the list of building that are damaged

                Damage\_year(:,:,yearNext)=Damage\_storm(:,:,StormIDsInYear);

            end

        end

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %

        %           Mitigation

        %

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        %Case 1 (return to original resistance level)

        %Damage is recorded above; no need to change resistance level in Parcel

        %Case 2 (probability of some upgrade)

        %If parcel experienced damage at some point during 'year', they will upgrade

        %     with some probability

        if caseNo==2

            if ~isnan(StormIDsInYear)

                [Parcel, damageTemp,Upgrade\_year\_tracker] = ...

Mitigation\_Scenario2\_Upgrade(Damage\_year(:,:,yearNext), Parcel, ...

ProbMit, Upgrade\_year\_tracker, year);

                Damage\_year(:,:,yearNext) = damageTemp;

            end

        end

        %Case 3 (neighborhood effect)

        if caseNo==3

            if ~isnan(StormIDsInYear)

                [Parcel, damageTemp,Upgrade\_year\_tracker] = ...

                Mitigation\_Scenario2\_Upgrade(Damage\_year(:,:,yearNext), Parcel, ProbMit, ... Upgrade\_year\_tracker,year);

Damage\_year(:,:,yearNext) = damageTemp;

[Parcel,damageNeighTemp, Upgrade\_year\_tracker] = ...

Mitigation\_Scenario3\_Neighbor\_Mod(Parcel, Damage\_year(:,:,yearNext), ...

threshold\_Neigh, year, Upgrade\_year\_tracker, Neigh\_List);

 Damage\_year(:,:,yearNext) = damageNeighTemp;

            end

        end

        %Case 4 (tax incentive - x% of home upgrade to next resistance level)

        if caseNo==4

            [Parcel, Upgrade\_year\_tracker] = Mitigation\_Scenario4\_Policy(Parcel, ... percentHouseGivenSubsidy, year,Upgrade\_year\_tracker);

    end

    %for all cases, count number of upgrades for each parcel

    NoUpgradesMaster = CountNumUpgrades\_inReplication(Parcel, Parcel\_Input, Parcel\_forUpgadeCount);

    %add Parcel and Damage for replication i to master Parcel and Damage

    Damage\_Master\_Year= Damage\_year;

    Upgrade\_year\_tracker\_Rep = Upgrade\_year\_tracker;

    %Save data

    save\_data(replication, city, caseNo, Damage\_Master\_Year, Upgrade\_year\_tracker\_Rep, ...

        NoUpgradesMaster, Upgrade\_Sensitivity\_factor, threshold\_Neigh, percentHouseGivenSubsidy)

    %organize damage data and save

    Extract\_Output\_Data(city, numHurr, Damage\_Master\_Year, replication,...

        caseNo, Upgrade\_Sensitivity\_factor, percentHouseGivenSubsidy,...

        threshold\_Neigh);

end

if parallel

    delete(myPool);

end

consolidateOutputData\_fromBatches(city, numHurr, numReplications, caseNo, ...

Upgrade\_Sensitivity\_factor, percentHouseGivenSubsidy, threshold\_Neigh)

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(2) Return\_Damage\_Initial.m**

function Damage\_Internal = Return\_Damage\_Initial(Parcel,Year)

Damage\_Internal = zeros(size(Parcel,1),12);

Damage\_Internal(:,1) = Year;

Damage\_Internal(:,2:5) = Parcel(:,1:4);

Damage\_Internal(:,6) = Parcel(:,4);   %Resistance level new

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(3) findDamageGivenWS.m**

function [Damage\_Internal] = findDamageGivenWS(Parcel, Fragility\_Curves, Windspeed, Year)

% This function probabilistically assigns damage states (0, 1, 2, 3, or 4) given wind speed for all parcels

%Called by: Hazard\_SEES\_ABM.m

%%%%%%%OUTPUT%%%%%%%%%%%%%

%Damage\_Internal: a matrix (double) that lists all parcels and their damage

% if any in 'Year'; size: noParcel(active+old) x 9 matrix (double)

%     Col1: year

%     Col2: Parcel ID

%     Col3: Parcel active? NaN if yes or ## if superceeded

% Col4: Building Type (1-11),

%     Col5: Resistance Level at START of year

%     Col6: Resistance Level at END of year

%     Col7: Max Damage State (1-4) in year

%     Col8: Windspeed that produced damage in year

%Define the windspeed for fragility curves

AllWindspeed= linspace(50,250,41);

%This is a unit converter; 1.151 is knots --> mph

WS\_converter = 1.151;

%inialize output 'Damage\_Interal'; is the number of Parcels (active + old)

Damage\_Internal = zeros(size(Parcel,1),8);

Damage\_Internal(:,1) = Year;

Damage\_Internal(:,2:5) = Parcel(:,1:4);

Damage\_Internal(:,6) = Parcel(:,4);   %Resistance level new (same as old until updated)

%for each parcel (given its building type and resistance level,

%    determine if there is damage and the severity of damage

for BuildingType = 1:11  %11 building types

    for ResistanceLevel = 1:size(Fragility\_Curves{BuildingType,1},1)

        %list of active parcels with building type i and resistance level j

        Parcel\_BuildingResistance\_IDs =  Parcel(Parcel(:,3)==...

            BuildingType & Parcel(:,4)==ResistanceLevel & isnan(Parcel(:,2)),1);

        %What are the rows in Parcel that correspond to the Parcel\_BuildingResistance\_IDs list?

        Parcel\_BuildingResistance\_IDs\_index = find(Parcel(:,3)==...

            BuildingType&Parcel(:,4)==ResistanceLevel&isnan(Parcel(:,2))); %the indices in

        %Parcel that contain parcels with BT, RL, and that are active

        %setOfColumns: find columns in Windspeed that refer to Parcel\_BuildingResistance\_IDs

        setOfColumns = ismember(Windspeed(1,:),Parcel\_BuildingResistance\_IDs);

 %ParcelWS: Array of Parcel windspeed, in mph (for 'BuildingType' and 'ResistanceLevel')

        ParcelWS = Windspeed(2, setOfColumns)\*WS\_converter;

        %Prob of DS\_X given windspeed, building type, and building resistance level

        ProbDamageState\_1= interp1(AllWindspeed, ...

 Fragility\_Curves{BuildingType,1}(ResistanceLevel,:,1), ParcelWS);

ProbDamageState\_2= interp1(AllWindspeed, ... Fragility\_Curves{BuildingType,1}(ResistanceLevel,:,2), ParcelWS);

ProbDamageState\_3= interp1(AllWindspeed, ... Fragility\_Curves{BuildingType,1}(ResistanceLevel,:,3), ParcelWS);

ProbDamageState\_4= interp1(AllWindspeed, ...

Fragility\_Curves{BuildingType,1}(ResistanceLevel,:,4), ParcelWS);

        %if isnan, convert Probability to 0

        ProbDamageState\_1(1,isnan(ProbDamageState\_1(1,:)))=0;

        ProbDamageState\_2(1,isnan(ProbDamageState\_2(1,:)))=0;

        ProbDamageState\_3(1,isnan(ProbDamageState\_3(1,:)))=0;

        ProbDamageState\_4(1,isnan(ProbDamageState\_4(1,:)))=0;

        %order ProbDamageState 1 through 4; This makes sure that the Prob of

        %being in each of the damage states goes in descending order

        temp = [ones(1,size(ProbDamageState\_1,2)); ProbDamageState\_1;ProbDamageState\_2;...

            ProbDamageState\_3;ProbDamageState\_4;zeros(1,size(ProbDamageState\_1,2))];

[~, DamageIndices] = sort(temp,'descend');

        ProbDamageState\_4(1,DamageIndices(5,:) ~= 5)=0;

        temp = [ones(1,size(ProbDamageState\_1,2)); ProbDamageState\_1;ProbDamageState\_2;...

            ProbDamageState\_3;ProbDamageState\_4;zeros(1,size(ProbDamageState\_1,2))];

        ProbDamageState = sort(temp,'descend');

        RandNo = rand(1,size(ProbDamageState\_1,2));

        DS = zeros(1,size(ProbDamageState\_1,2)); %No buildings of BuildingType\_i and ResLevel\_j,

%for each building with ResLevelj and BuildTypei and active

        for k= 1:size(ProbDamageState\_1,2)

            DS(1,k) = find(ProbDamageState(:,k)>RandNo(1,k),1,'last')-1;

            Parcel\_Index = Parcel\_BuildingResistance\_IDs\_index(k,1);

            %update Damage List, Parcel List if there is damage

            Damage\_Internal(Parcel\_Index,7:8) = [DS(1,k) ParcelWS(1,k)];

        end

    end

end

Damage\_Internal = Damage\_Internal(:,7:8);

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(4) Mitigation\_Scenario2\_Upgrade.m**

function [Parcel, Damage\_year,Upgrade\_year\_tracker] = Mitigation\_Scenario2\_Upgrade(Damage\_year, Parcel, ProbMit,Upgrade\_year\_tracker,year)

% This function is for mitigation Option 2 - when a house is damaged after a

% storm, there is some likelihood that it will increase its resistance level

% given its damage state and the resistance level in which it began

%

%Called by: Hazard\_SEES\_ABM.m

for i = 1:size(Damage\_year,1) %number of Parcels

      if isnan(Damage\_year(i,3))

            DamageLevel = Damage\_year(i,7);  %House Damage Level

            if DamageLevel>=1

                  %House Resistance Level before storm

                  ResistanceLevelCurrent  = Damage\_year(i,5);

                  BuildingType = Damage\_year(i,4);

                  if BuildingType<6

                      Type = 1;

                  elseif BuildingType<10

                      Type = 2;

                  elseif BuildingType==10

                      Type = 3;

                  else

                      Type = 4;

                  end

 %Probability of upgradeS given resistance

                  Prob = ProbMit{Type,1}(:,DamageLevel, ResistanceLevelCurrent);

                  Prob = cumsum(Prob);

                  RandNo = rand();

                  %find index in Prob where RandNo lies <- this the is new

                  %resistance level

                  temp= RandNo-Prob;

                  NewResistanceLevel = find(temp<=0,1);

                  if BuildingType==10 && NewResistanceLevel>2

                      NewResistanceLevel=2;

                  end

                  %Update Parcel information

                  Damage\_year(i,6) = NewResistanceLevel; %col 6: new resistance level

                  Parcel(i,4) = NewResistanceLevel;

                  temp2 = find(Upgrade\_year\_tracker(Parcel(i,1),:)>0);

                  if temp2(end) < 10 && NewResistanceLevel > ResistanceLevelCurrent

                      Upgrade\_year\_tracker(Parcel(i,1),temp2(end)+1) = year; %year

                      Upgrade\_year\_tracker(Parcel(i,1),temp2(end)+2) = NewResistanceLevel; %new

                  end

            end

      end

end

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(5) Mitigation\_Scenario3\_Neighbor\_Mod.m**

 function [Parcel, Damage\_year, Upgrade\_year\_tracker] = Mitigation\_Scenario3\_Neighbor\_Mod(Parcel, ...

    Damage\_year, threshold\_Neigh,year, Upgrade\_year\_tracker, Neigh\_List)

%This function is for mitigation Option 3 - after a storm, there is some likelihood

%that an undamaged house will increase its resistance level if some percentage of neighbors

%experience damage. This is run after case 2\_Base.

%Rule: for a house the is undamaged, if the more than threshold\_Neigh% of the

%      neighbors (i.e., houses within 75m) are damaged in 'year', they will

%      improve the construction quality one resistance level

%Called by: Hazard\_SEES\_ABM.m

%Calls: Returns\_NewResistanceLevel.m

%Input:

%Upgrade\_year\_tracker: Counts the number of upgrades that are made each year

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%Find the list of active parcels that are neighbors with a damaged house but

%are themselves not damaged

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%find the neighbors (ID) of damaged houses who themselves are not damaged

ParcelIDs\_damaged = Damage\_year(Damage\_year(:,7)>1,1);

NeighborToDamagedParcel = zeros(size(Damage\_year,1),1);

next=1;

for i = 1:size(ParcelIDs\_damaged,1) %add neighbors of ParcelIDs\_damaged(i) to list if they are not

 damaged

    Neigh=Neigh\_List(ParcelIDs\_damaged(i),Neigh\_List(ParcelIDs\_damaged(i),:)>0);

    for j = 1:size(Neigh,2)

        if ~ismember(Neigh(j),ParcelIDs\_damaged) && ~ismember(Neigh(j),NeighborToDamagedParcel)

%is neigh(i) is damaged, if yes, don't add to list

            NeighborToDamagedParcel(next,1) = Neigh(j);

            next=next+1;

        end

    end

end

NeighborToDamagedParcel = NeighborToDamagedParcel(NeighborToDamagedParcel(:,1)>0,1);

for i =1:size(NeighborToDamagedParcel,1)

    %determine how many neighbors to i that are damaged

    Neigh = Neigh\_List(NeighborToDamagedParcel(i),Neigh\_List(NeighborToDamagedParcel(i),:)>0);

    NumNeigh = size(Neigh,2);

    NumDam2 = 0;

    %find Neigh that are damaged

    for j = 1:size(Neigh,2)

        if Damage\_year(Neigh(j),7)>1

          NumDam2=NumDam2+1;

        end

    end

    BuildingType = Parcel(NeighborToDamagedParcel(i),3);

    ResisLevel\_current = Parcel(NeighborToDamagedParcel(i),4);

    if NumDam2/NumNeigh > threshold\_Neigh %0.5

       temp = Returns\_NewResistanceLevel(ResisLevel\_current,BuildingType);

       Parcel(NeighborToDamagedParcel(i),4) = temp;

       Damage\_year(NeighborToDamagedParcel(i),7) = temp;

       temp2 = find(Upgrade\_year\_tracker(NeighborToDamagedParcel(i,1),:)>0);

       if temp2(end)<10

           Upgrade\_year\_tracker(NeighborToDamagedParcel(i,1),temp2(end)+1) = year;

           Upgrade\_year\_tracker(NeighborToDamagedParcel(i,1),temp2(end)+2) = temp;

       end

    end

end

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(6) Mitigation\_Scenario4\_Policy.m**

function [Parcel,Upgrade\_year\_tracker] = Mitigation\_Scenario4\_Policy(Parcel, percentHouseGivenSubsidy, year,Upgrade\_year\_tracker)

% This function is for mitigation Option 4 - some percentage of homes (regardless of damage) are given % subsidies to upgrade. Subsidies are not given to houses that are in resistance level 7 (max).

%Called by: Hazard\_SEES\_ABM.m

%Parcels that are active and have at resistance level of 6 or less. A double.

Parcels\_Active=Parcel(isnan(Parcel(:,2)),:); %active parcels

Parcel\_subset\_IDs\_NaN7 = Parcels\_Active(Parcels\_Active(:,4)<Parcels\_Active(:,6),1);

%Number of active parcels, parcel given subsidy, number of parcels eligble for subsidy

NumParcels = size(Parcels\_Active,1); %number of active parcels; includes houses with RL 7

NumParcelsGivenSubsidy =round(percentHouseGivenSubsidy\*NumParcels);

NumParcels\_NaN7 = size(Parcel\_subset\_IDs\_NaN7,1); %num active parcels eligible for the subsidy

if NumParcels\_NaN7<NumParcelsGivenSubsidy   %number of houses that are active and with RL less

      %     than 7 is fewer than the x% choosen by the government for an upgrade.

      NumParcelsGivenSubsidy=NumParcels\_NaN7;

elseif NumParcels\_NaN7==0  %number of houses that are active and with RL less than 7 is 0

      %ParcelIDs\_GivenSubsidy=NaN;

      return; %No parcel is eligible for a subsidy

end

ParcelIDs\_GivenSubsidy = Parcel\_subset\_IDs\_NaN7(randperm(NumParcels\_NaN7,NumParcelsGivenSubsidy),1);

      %randomly choose NumParcelsGivenSubsidy in NumParcels\_NaN7 list for an upgrade

for i = 1:NumParcelsGivenSubsidy

      ResisLevel\_current = Parcel(ParcelIDs\_GivenSubsidy(i,1),4);

      BuildingType = Parcel(ParcelIDs\_GivenSubsidy(i,1),3);

      temp= Returns\_NewResistanceLevel(ResisLevel\_current,BuildingType);

      Parcel(ParcelIDs\_GivenSubsidy(i,1),4) =temp;

      temp2 = find(Upgrade\_year\_tracker(ParcelIDs\_GivenSubsidy(i,1),:)>0);

      if temp2(end) < 10

          Upgrade\_year\_tracker(ParcelIDs\_GivenSubsidy(i,1),temp2(end)+1) = year;

          Upgrade\_year\_tracker(ParcelIDs\_GivenSubsidy(i,1),temp2(end)+2) = temp;

      end

end

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(7) Returns\_NewResistanceLevel.m**

function NewResistanceLevel =  Returns\_NewResistanceLevel(ResistanceLevel, BuildingType)

%Called by  Mitigation\_Option3.m

%           Mitigation\_Option4.m

if BuildingType < 6

    if ResistanceLevel==1

        NewResistanceLevel = 2;

    elseif  ResistanceLevel==2

        y = rand();

        if y>0.5

            NewResistanceLevel = 3;

        else

            NewResistanceLevel = 4;

        end

    elseif ResistanceLevel==3

        NewResistanceLevel = 5;

    elseif  ResistanceLevel==4

        NewResistanceLevel = 5;

    elseif  ResistanceLevel==5

        NewResistanceLevel = 6;

    else

        NewResistanceLevel = 6;

    end

elseif BuildingType < 10

    if ResistanceLevel==1

        NewResistanceLevel = 2;

    elseif  ResistanceLevel==2

        y = rand();

        if y>0.5

            NewResistanceLevel = 3;

        else

            NewResistanceLevel = 4;

        end

    elseif  ResistanceLevel==3

        NewResistanceLevel = 5;

    elseif  ResistanceLevel==4

        NewResistanceLevel = 6;

    elseif  ResistanceLevel==6

        NewResistanceLevel = 7;

    elseif  ResistanceLevel==5

        NewResistanceLevel = 7;

    else

        NewResistanceLevel=7;

    end

end

if BuildingType == 10

    if ResistanceLevel==1

        NewResistanceLevel = 2;

    else

        NewResistanceLevel = 2;

    end

end

if BuildingType == 11

    if ResistanceLevel==1

        NewResistanceLevel = 2;

    else

        NewResistanceLevel = 3;

    end

end

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(8) CountNumUpgrades\_inReplication.m**

function NoUpgradesMaster = CountNumUpgrades\_inReplication(Parcel, Parcel\_Input, Parcel\_forUpgadeCount)

numParcelsInitial = size(Parcel,1);

temp = Parcel(:,4)-Parcel\_Input(:,4);

for index\_Upgrade = 1:numParcelsInitial

    if Parcel\_Input(index\_Upgrade,3) < 6 && temp(index\_Upgrade,1)>1

        if temp(index\_Upgrade,1)>=2 && Parcel\_forUpgadeCount(index\_Upgrade,4)==2

            temp(index\_Upgrade,1) = temp(index\_Upgrade,1) - 1;

        elseif temp(index\_Upgrade,1)>=2 && Parcel\_forUpgadeCount(index\_Upgrade,4)==3

            temp(index\_Upgrade,1) = temp(index\_Upgrade,1) - 1;

        end

    elseif Parcel\_Input(index\_Upgrade,3) < 10 && temp(index\_Upgrade,1)>1

        if temp(index\_Upgrade,1)==2 && Parcel\_forUpgadeCount(index\_Upgrade,4)==2

            temp(index\_Upgrade,1) = temp(index\_Upgrade,1) - 1;

        elseif temp(index\_Upgrade,1)>2 && Parcel\_forUpgadeCount(index\_Upgrade,4)==2

            temp(index\_Upgrade,1) = temp(index\_Upgrade,1) - 2;

        elseif temp(index\_Upgrade,1)==2 && Parcel\_forUpgadeCount(index\_Upgrade,4)==3

            temp(index\_Upgrade,1) = temp(index\_Upgrade,1) - 1;

        elseif temp(index\_Upgrade,1)>2 && Parcel\_forUpgadeCount(index\_Upgrade,4)==3

            temp(index\_Upgrade,1) = temp(index\_Upgrade,1) - 2;

        end

    end

end

NoUpgradesMaster = temp;

end

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

**(9) save\_data.m**

function save\_data(replication, city, caseNo, Damage\_Master\_Year, Upgrade\_year\_tracker\_Rep, ...

    NoUpgradesMaster, Upgrade\_Sensitivity\_factor, threshold\_Neigh, percentHouseGivenSubsidy)

if caseNo==1

    X = strcat('Damage\_Master\_',city,'\_Baseline\_',int2str(replication),'.mat');

    parsave\_Damage(X, Damage\_Master\_Year);

    Y = strcat('NoUpgradesMaster\_',city,'\_Baseline\_',int2str(replication),'.mat');

    parsave\_UpgradeNo(Y,NoUpgradesMaster,Upgrade\_year\_tracker\_Rep);

elseif caseNo==3 && Upgrade\_Sensitivity\_factor >= 0

    X = strcat('Damage\_Master\_',city,'\_NeighborCase\_USensF',int2str(Upgrade\_Sensitivity\_factor\*100),...

'\_NeighT',int2str(threshold\_Neigh\*100),'\_',int2str(replication),'.mat');

    parsave\_Damage(X, Damage\_Master\_Year);

    Y=strcat('NoUpgradesMaster\_',city,'\_NeighborCase\_USens',int2str(Upgrade\_Sensitivity\_factor\*100),...

'\_NeighT',int2str(threshold\_Neigh\*100),'\_',int2str(replication),'.mat');

    parsave\_UpgradeNo(Y,NoUpgradesMaster,Upgrade\_year\_tracker\_Rep);

elseif caseNo==3 && Upgrade\_Sensitivity\_factor < 0

    X =strcat('Damage\_Master\_',city,'\_NeighborCase\_USensFm',int2str(Upgrade\_Sensitivity\_factor\*100),...

'\_NeighT', int2str(threshold\_Neigh\*100),'\_',int2str(replication),'.mat');

 parsave\_Damage(X, Damage\_Master\_Year);

    Y = strcat('NoUpgradesMaster\_',city,'\_NeighborCase\_USensFm',...

int2str(Upgrade\_Sensitivity\_factor\*100),'\_NeighT',int2str(threshold\_Neigh\*100),...

'\_',int2str(replication),'.mat');

    parsave\_UpgradeNo(Y,NoUpgradesMaster,Upgrade\_year\_tracker\_Rep);

elseif caseNo==2 && Upgrade\_Sensitivity\_factor >= 0

    X = strcat('Damage\_Master\_',city,'\_UpgradeCase\_',int2str(Upgrade\_Sensitivity\_factor\*100),...

'\_',int2str(replication),'.mat');

    parsave\_Damage(X, Damage\_Master\_Year);

    Y = strcat('NoUpgradesMaster\_',city,'\_UpgradeCase\_',int2str(Upgrade\_Sensitivity\_factor\*100),...

'\_',int2str(replication),'.mat');

    parsave\_UpgradeNo(Y,NoUpgradesMaster,Upgrade\_year\_tracker\_Rep);

elseif caseNo==2 && Upgrade\_Sensitivity\_factor < 0

    X = strcat('Damage\_Master\_',city,'\_UpgradeCase\_m',int2str(Upgrade\_Sensitivity\_factor\*100),...

'\_',int2str(replication),'.mat');

    parsave\_Damage(X, Damage\_Master\_Year);

    Y = strcat('NoUpgradesMaster\_',city,'\_UpgradeCase\_m',int2str(Upgrade\_Sensitivity\_factor\*100),...

'\_',int2str(replication),'.mat');

    parsave\_UpgradeNo(Y,NoUpgradesMaster,Upgrade\_year\_tracker\_Rep);

elseif caseNo ==4

    X = strcat('Damage\_Master\_',city,'\_PolicyCase\_',int2str(percentHouseGivenSubsidy\*100),...

'\_',int2str(replication),'.mat');

    parsave\_Damage(X, Damage\_Master\_Year);

    Y = strcat('NoUpgradesMaster\_',city,'\_PolicyCase\_',int2str(percentHouseGivenSubsidy\*100),...

'\_',int2str(replication),'.mat');

    parsave\_UpgradeNo(Y,NoUpgradesMaster,Upgrade\_year\_tracker\_Rep);

end

end