Example: Proton Treatment Plan with Manipulated CT values

Table of Contents

%%%

%%%

In this example we will show (i) how to load patient data into matRad (ii) how to setup a proton dose calculation (iii) how to inversely optimize the pencil beam intensities directly from command window in MATLAB. (iv) how to re-optimize a treatment plan (v) how to manipulate the CT cube by adding noise to the cube (vi) how to recalculate the dose considering the manipulated CT cube and the previously optimized pencil beam intensities (vii) how to compare the two results

Patient Data Import

Let's begin with a clear Matlab environment and import the prostate patient into your workspace.

```
clc,clear,close all;
load('PROSTATE.mat');
```

Treatment Plan

The next step is to define your treatment plan labeled as 'pln'. This structure requires input from the treatment planner and defines the most important cornerstones of your treatment plan.

```
pln.radiationMode            = 'protons';
pln.machine                  = 'generic_MCsquare';
pln.numOfFractions           = 30;
pln.propOpt.bioOptimization = 'const_RBExD';
pln.propStf.gantryAngles    = [90 270];
pln.propStf.couchAngles     = [0 0];
pln.propStf.bixelWidth      = 3;
pln.propStf.numOfBeams      = numel(pln.propStf.gantryAngles);
pln.propStf.isoCenter       = ones(pln.propStf.numOfBeams,1) *
 matRad_getIsoCenter(cst,ct,0);
pln.propOpt.runDAO          = 0;
pln.propOpt.runSequencing   = 0;

% dose calculation settings
pln.propDoseCalc.doseGrid.resolution.x = 3; % [mm]
pln.propDoseCalc.doseGrid.resolution.y = 3; % [mm]
pln.propDoseCalc.doseGrid.resolution.z = 3; % [mm]
```

Generate Beam Geometry STF

```
stf = matRad_generateStf(ct,cst,pln);
```

```
matRad: Generating stf struct... Warning: Could not find HLUT
Philips-AcQSimCT-ConvolutionKernel-000000_protons.hlut in hlutLibrary
 folder.
matRad default HLUT loaded
Progress: 100.00 %
```

Dose Calculation

```
dij = matRad_calcParticleDose(ct,stf,pln,cst);
```

```
Warning: Could not find HLUT
Philips-AcQSimCT-ConvolutionKernel-000000_protons.hlut in hlutLibrary
 folder.
matRad default HLUT loaded
matRad: Using a constant RBE of 1.1
matRad: Particle dose calculation...
Beam 1 of 2:
matRad: calculate radiological depth cube...done.
matRad: calculate lateral cutoff...done.
Progress: 59.80 %

Requested 2207926851x8 (16.5GB) array exceeds maximum array size
 preference. Creation of arrays greater than this limit may take
 a long time and cause MATLAB to become unresponsive. See <a
 href="matlab: helpview([docroot '/matlab/helptargets.map'],
 'matlab_env_workspace_prefs')">array size limit</a> or preference
 panel for more information.

Error in matRad_calcDoseFillDij (line 30)
        dij.physicalDose{1}(:,(ceil(counter/
numOfBixelsContainer)-1)*numOfBixelsContainer+1:counter) =
 [doseTmpContainer{1:mod(counter-1,numOfBixelsContainer)+1,1}];

Error in matRad_calcParticleDose (line 288)
```

```
                matRad_calcDoseFillDij;

Error in matRad_example6_protonsNoise (line 58)
dij = matRad_calcParticleDose(ct,stf,pln,cst);
```

Inverse Optimization for IMPT

```
resultGUI = matRad_fluenceOptimization(dij,cst,pln);
```

Calculate quality indicators

```
[dvh,qi]       = matRad_indicatorWrapper(cst,pln,resultGUI);
ixRectum       = 1;
display(qi(ixRectum).D_5);
```

Let's change the optimization parameter of the rectum in such a way that it will be better spared. We increase the penalty and lower the threshold of the squared overdose objective function. Afterwards we re-optimize the treatment plan and evaluate dose statistics one more time.

```
objective = cst{ixRectum,6}{1}; %This gives a struct
objective =
 matRad_DoseOptimizationFunction.createInstanceFromStruct(objective); %Now
 we turn it into a class
objective = objective.setDoseParameters(40); %We can simply call this
 function to change the/all dose parameter(s)
cst{ixRectum,6}{1} = struct(objective); % We put it back as struct

cst{ixRectum,6}{1}.parameters{1} = 40;
cst{ixRectum,6}{1}.penalty = 500;
resultGUI                = matRad_fluenceOptimization(dij,cst,pln);
[dvh2,qi2]               = matRad_indicatorWrapper(cst,pln,resultGUI);
display(qi2(ixRectum).D_5);
```

Plot the Resulting Dose Slice

Let's plot the transversal iso-center dose slice

```
slice = round(pln.propStf.isoCenter(1,3)./ct.resolution.z);
figure
imagesc(resultGUI.RBExDose(:,:,slice)),colorbar, colormap(jet)
```

Now let's simulate a range undershoot by scaling the relative stopping power cube by 3.5% percent

```
ct_manip         = ct;
ct_manip.cubeHU{1} = 1.035*ct_manip.cubeHU{1};
```

Recalculate Plan with MC square

Let's use the existing optimized pencil beam weights and recalculate the RBE weighted dose

```
resultGUI_noise =
 matRad_calcDoseDirectMC(ct_manip,stf,pln,cst,resultGUI.w);
```

Visual Comparison of results

Let's compare the new recalculation against the optimization result.

```
plane      = 3;
doseWindow = [0 max([resultGUI.RBExDose(:);
 resultGUI_noise.RBExDose(:)])];

figure,title('original plan')
matRad_plotSliceWrapper(gca,ct,cst,1,resultGUI.RBExDose,plane,slice,
[],0.75,colorcube,[],doseWindow,[]);
figure,title('manipulated plan')
matRad_plotSliceWrapper(gca,ct_manip,cst,1,resultGUI_noise.RBExDose,plane,slice,
[],0.75,colorcube,[],doseWindow,[]);

% Let's plot single profiles along the beam direction
ixProfileY = round(pln.propStf.isoCenter(1,1)./ct.resolution.x);

profileOrginal = resultGUI.RBExDose(:,ixProfileY,slice);
profileNoise   = resultGUI_noise.RBExDose(:,ixProfileY,slice);

figure,plot(profileOrginal,'LineWidth',2),grid on,hold on,
       plot(profileNoise,'LineWidth',2),legend({'original
 profile','noise profile'}),
       xlabel('mm'),ylabel('Gy(RBE)'),title('profile plot')
```

Quantitative Comparison of results

Compare the two dose cubes using a gamma-index analysis.

```
doseDifference   = 2;
distToAgreement  = 2;
n                = 1;

[gammaCube,gammaPassRateCell] = matRad_gammaIndex(...
    resultGUI_noise.RBExDose,resultGUI.RBExDose,...
    [ct.resolution.x, ct.resolution.y, ct.resolution.z],...
    [doseDifference distToAgreement],slice,n,'global',cst);
```

Published with MATLAB® R2018a