Example: Carbon Ion Treatment Plan

Table of Contents

In this example we will show (i) how to load patient data into matRad (ii) how to setup a carbon ion dose calculation plan including variable RBE optimization (iii) how to inversely optimize the pencil beam intensities (v) how to change the tissues radiobiological characteristics (vi) how to recalculated the dose considering the previously optimized pencil beam intensities (vii) how to compare the two results

Patient Data Import

Let's begin with a clear Matlab environment. First, import the liver patient into your workspace. The phantom is comprised of a 'ct' and 'cst' structure defining the CT images and the structure set. Make sure the matRad root directory with all its SUBDIRECTORIES is added to the Matlab search path.

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

Let's check the two variables, we have just imported. First, the 'ct' variable comprises the ct cube along with some meta information describing properties of the ct cube (cube dimensions, resolution, number of CT scenarios). Please note that multiple ct cubes (e.g. 4D CT) can be stored in the cell array ct.cube{}

```
ct
```

```
ct =

  struct with fields:
         cube: {[217×217×168 double]}
   resolution: [1×1 struct]
      cubeDim: [217 217 168]
  numOfCtScen: 1
```

The 'cst' cell array defines volumes of interests along with information required for optimization. Each row belongs to one certain VOI, whereas each column defines different properties. Specifically, the second and third column show the name and the type of the structure. The type can be set to OAR, TARGET or IGNORED. The fourth column depicts a linear index vector depicting voxels in the CT cube that are covered by the corresponding VOI. In total, 17 structures are defined in the cst

```
cst
```

```
cst =

  17×6 cell array

  Columns 1 through 5

    [ 0]    'CTV'            'TARGET'     {1×1 cell}    [1×1 struct]
    [ 1]    'Celiac'         'OAR'        {1×1 cell}    [1×1 struct]
    [ 2]    'DoseFalloff'    'OAR'        {1×1 cell}    [1×1 struct]
    [ 3]    'GTV'            'TARGET'     {1×1 cell}    [1×1 struct]
    [ 4]    'Heart'          'OAR'        {1×1 cell}    [1×1 struct]
    [ 5]    'KidneyL'        'OAR'        {1×1 cell}    [1×1 struct]
    [ 6]    'KidneyR'        'OAR'        {1×1 cell}    [1×1 struct]
    [ 7]    'LargeBowel'     'OAR'        {1×1 cell}    [1×1 struct]
    [ 8]    'Liver'          'OAR'        {1×1 cell}    [1×1 struct]
    [ 9]    'PTV'            'TARGET'     {1×1 cell}    [1×1 struct]
    [10]    'SMASMV'         'OAR'        {1×1 cell}    [1×1 struct]
    [11]    'Skin'           'OAR'        {1×1 cell}    [1×1 struct]
    [12]    'SmallBowel'     'OAR'        {1×1 cell}    [1×1 struct]
    [13]    'SpinalCord'     'OAR'        {1×1 cell}    [1×1 struct]
    [14]    'Stomach'        'IGNORED'    {1×1 cell}    [1×1 struct]
    [15]    'duodenum'       'OAR'        {1×1 cell}    [1×1 struct]
    [16]    'entrance'       'IGNORED'    {1×1 cell}    [1×1 struct]

  Column 6

              []
              []
              []
              []
              []
              []
              []
              []
              []
    [1×1 struct]
              []
    [1×1 struct]
              []
              []
              []
              []
              []
```

The fifth column represents meta parameters used for optimization such as the overlap priority, which can be specified in double precision. A lower overlap priority indicates increased importance. In contrast, a higher overlap priority indicates a structure with lower importance. The parameters alphaX and betaX depict the tissue's photon radio sensitivity parameter of the linear quadratic model. These parameter are required for biological treatment planning using a variable RBE. Since the PTV, is stored in the tenth column, lets output the PTVs meta information

```
cst{10,5}
```

```
ans = 

  struct with fields:

    TissueClass: 1
         alphaX: 0.1000
          betaX: 0.0500
       Priority: 1
        Visible: 1
```

The sixth column contains optimization information such as objectives and constraints which are required to calculate the objective function value. Please note, that multiple objectives/constraints can be defined for individual structures. Here, we have defined a squared deviation objective for the target thereby equally penalizing over and underdosage.

```
cst{10,6}
```

```
ans = 

  struct with fields:

          type: 'square deviation'
       penalty: 100
          dose: 45
           EUD: NaN
        volume: NaN
    robustness: 'none'
```

Now lets increase the importance of the squared deviation objective to 2000

```
cst{10,6}.penalty = 2000;
```

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.

First of all, we need to define what kind of radiation modality we would like to use. Possible values are photons, protons or carbon. In this example we would like to use carbon ions for treatment planning. Then, we need to define a treatment machine to correctly load the corresponding base data. Since we provide generic base data we set the machine to 'Genereric. By this means matRad will look for 'proton_Generic.mat' in our root directory and will use the data provided in there for dose calculation

```
pln.radiationMode = 'carbon';
pln.machine       = 'Generic';
```

Define the flavor of biological optimization for treatment planning along with the quantity that should be used for optimization. Possible values are (none: physical optimization; const_RBExD: constant RBE of 1.1; LEMIV_effect: effect-based optimization; LEMIV_RBExD: optimization of RBE-weighted dose. As we use carbon ions, we what to use base data from the local effect model IV and want to optimize the RBE-weighted dose. Therefore we set bioOptimization to LEMIV_RBExD

```
pln.bioOptimization = 'LEMIV_RBExD';
```

Now we have to set some beam parameters. We can define multiple beam angles for the treatment and pass these to the plan as a vector. matRad will then interpret the vector as multiple beams. However, now we define a single beam with a gantry angle of 315 degree and set the corresponding couch angle to 0 degree. Furthermore, we want the lateral pencil beam spacing in x and y to be 3 mm in the iso-center. In total we are using 30 fractions. It is noteworthy that matRad is always optimizing the fraction dose.

```
pln.gantryAngles    = 315;
pln.couchAngles     = 0;
pln.bixelWidth      = 3;
pln.numOfFractions  = 30;
```

Obtain the number of beams and voxels from the existing variables and calculate the iso-center which is per default the mass of gravity of all target voxels.

```
pln.numOfBeams      = numel(pln.gantryAngles);
pln.numOfVoxels     = prod(ct.cubeDim);
pln.voxelDimensions = ct.cubeDim;
pln.isoCenter       = ones(pln.numOfBeams,1) *
 matRad_getIsoCenter(cst,ct,0);
```

Disable sequencing and direct aperture optimization, as we have a particle plan.

```
pln.runDAO        = 0;
pln.runSequencing = 0;
```

and schwuppdiwupp - our treatment plan is ready. Lets have a look at it:

```
pln
```

```

pln = 

  struct with fields:

      radiationMode: 'carbon'
            machine: 'Generic'
    bioOptimization: 'LEMIV_RBExD'
       gantryAngles: 315
        couchAngles: 0
         bixelWidth: 3
```

```
     numOfFractions: 30
         numOfBeams: 1
        numOfVoxels: 7910952
    voxelDimensions: [217 217 168]
          isoCenter: [265.5030 296.4758 316.4222]
             runDAO: 0
      runSequencing: 0
```

Generate Beam Geometry STF

This acronym stands for steering file and comprises the complete beam geometry along with ray position, pencil beam positions and energies, source to axis distance (SAD) etc.

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

```
matRad: Generating stf struct... Progress: 100.00 %
```

Let's display the beam geometry information

```
stf
```

```
stf = 

  struct with fields:

          gantryAngle: 315
           couchAngle: 0
           bixelWidth: 3
        radiationMode: 'carbon'
                  SAD: 10000
            isoCenter: [265.5030 296.4758 316.4222]
            numOfRays: 482
                  ray: [1×482 struct]
       sourcePoint_bev: [0 -10000 0]
          sourcePoint: [-7.0711e+03 -7.0711e+03 0]
    numOfBixelsPerRay: [1×482 double]
     totalNumOfBixels: 11780
```

Output the total number of rays.

```
stf.numOfRays
```

```
ans =

   482
```

The following sub-structure stores the number of bixels/pencil beams per ray. Exemplary lets out put information of ray 100.

```
N = stf.numOfBixelsPerRay(100)
```

```
N =

     9
```

Let's have a closer look on the stf.ray sub-structure since it contains the actual beam/ray geometry information. For illustration purposes we want to show the one-hundreds ray. Besides geometrical information about the position and orientation of the ray, we can also find pencil beam information. If the ray coincides with the target, pencil beams were defined along the ray from target entry to target exit.

```
stf.ray(100)
```

```
ans =

  struct with fields:

         rayPos_bev: [-21 0 27]
    targetPoint_bev: [-42 10000 54]
             rayPos: [-14.8492 14.8492 27]
        targetPoint: [7.0414e+03 7.1008e+03 54]
             energy: [1×9 double]
            focusIx: [1 1 1 1 1 1 1 1 1]
       rangeShifter: [1×9 struct]
```

We now expect to find exactly N-different energy levels on the 100 rays.

```
stf.ray(100).energy
```

```
ans =

  Columns 1 through 7

  160.2600  163.3500  166.4100  169.4300  172.4100  175.3700  178.2800

  Columns 8 through 9

  181.1700  184.0300
```

Dose Calculation

Calculate dose influence matrix for unit pencil beam intensities.

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

```
matRad: loading biological base data... done.
matRad: Particle dose calculation...
Beam 1 of 1:
matRad: calculate radiological depth cube...done.
matRad: calculate lateral cutoff...done.
```

```
Progress: 100.00 %
```

Inverse Optimization for IMPT

The goal of the fluence optimization is to find a set of bixel/spot weights which yield the best possible dose distribution according to the clinical objectives and constraints underlying the radiation treatment.

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

```

******************************************************************************
This program contains Ipopt, a library for large-scale nonlinear
 optimization.
 Ipopt is released as open source code under the Eclipse Public
 License (EPL).
         For more information visit http://projects.coin-or.org/Ipopt
******************************************************************************

This is Ipopt version 3.11.8, running with linear solver ma57.

Number of nonzeros in equality constraint Jacobian...:        0
Number of nonzeros in inequality constraint Jacobian.:        0
Number of nonzeros in Lagrangian Hessian.............:        0

Total number of variables............................:    11780
                     variables with only lower bounds:    11780
                variables with lower and upper bounds:        0
                     variables with only upper bounds:        0
Total number of equality constraints.................:        0
Total number of inequality constraints...............:        0
        inequality constraints with only lower bounds:        0
   inequality constraints with lower and upper bounds:        0
        inequality constraints with only upper bounds:        0

iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
   0 5.9126463e+002 0.00e+000 1.27e+001   0.0 0.00e+000    -  0.00e
+000 0.00e+000   0
   1 4.2674568e+003 0.00e+000 1.30e+001  -0.5 6.35e-001    -
 8.18e-002 2.50e-001f  3
   2 5.9458027e+002 0.00e+000 6.25e+000  -1.2 7.77e-002    -  1.00e
+000 1.00e+000f  1
   3 7.7057128e+001 0.00e+000 5.34e+000  -1.2 7.27e-002    -  1.00e
+000 1.00e+000f  1
   4 5.2132707e+001 0.00e+000 1.64e+000  -1.5 2.86e-002    -  1.00e
+000 1.00e+000f  1
   5 2.4181948e+001 0.00e+000 7.10e-001  -3.3 9.13e-003    -  1.00e
+000 1.00e+000f  1
   6 1.8435155e+001 0.00e+000 6.36e-001  -4.1 7.06e-003    -  1.00e
+000 1.00e+000f  1
   7 1.4140149e+001 0.00e+000 4.63e-001  -4.8 8.97e-003    -  1.00e
+000 1.00e+000f  1
   8 9.9330268e+000 0.00e+000 4.67e-001  -4.8 1.53e-002    -  1.00e
+000 1.00e+000f  1
```

```
   9 7.6135863e+000 0.00e+000 5.28e-001  -4.1 2.47e-002    -  1.00e
+000 9.72e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  10 6.3088562e+000 0.00e+000 2.60e-001  -4.2 1.23e-002    -  1.00e
+000 7.70e-001f  1
  11 5.6519349e+000 0.00e+000 1.49e+000  -3.4 1.29e-002    -  1.00e
+000 1.00e+000f  1
  12 5.0092439e+000 0.00e+000 2.51e-001  -3.2 1.03e-002    -
 6.21e-001 1.00e+000f  1
  13 4.4129627e+000 0.00e+000 4.71e-001  -3.6 1.70e-002    -  1.00e
+000 7.42e-001f  1
  14 4.1243285e+000 0.00e+000 4.27e-001  -3.0 5.46e-003    -  1.00e
+000 1.00e+000f  1
  15 3.7506543e+000 0.00e+000 1.62e-001  -3.5 7.61e-003    -
 9.99e-001 1.00e+000f  1
  16 3.2921716e+000 0.00e+000 1.48e-001  -3.8 1.44e-002    -  1.00e
+000 9.90e-001f  1
  17 3.1260882e+000 0.00e+000 4.18e-001  -3.5 1.53e-002    -  1.00e
+000 6.49e-001f  1
  18 3.0310255e+000 0.00e+000 3.57e-001  -4.4 5.83e-003    -  1.00e
+000 2.72e-001f  1
  19 2.8473333e+000 0.00e+000 2.94e-001 -10.4 1.21e-002    -
 5.51e-001 3.23e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  20 2.7130315e+000 0.00e+000 8.26e-001  -3.7 1.69e-002    -
 6.05e-001 3.24e-001f  1
  21 2.6196092e+000 0.00e+000 3.24e-001  -5.9 1.17e-002    -
 5.22e-001 2.52e-001f  1
  22 2.4875011e+000 0.00e+000 1.89e-001  -4.3 1.33e-002    -
 9.72e-001 3.54e-001f  1
  23 2.3832875e+000 0.00e+000 7.30e-001  -4.0 1.37e-002    -  1.00e
+000 3.14e-001f  1
  24 2.3264291e+000 0.00e+000 4.23e-001 -10.0 1.39e-002    -
 4.94e-001 1.77e-001f  1
  25 2.2094192e+000 0.00e+000 3.60e-001  -4.7 1.80e-002    -  1.00e
+000 3.35e-001f  1
  26 2.1125782e+000 0.00e+000 4.80e-001  -4.0 2.14e-002    -
 9.99e-001 3.10e-001f  1
  27 2.0377549e+000 0.00e+000 6.24e-001  -3.3 1.19e-002    -
 5.98e-001 1.00e+000f  1
  28 1.9477063e+000 0.00e+000 2.16e-001  -4.3 1.21e-002    -
 6.73e-001 4.54e-001f  1
  29 1.8776445e+000 0.00e+000 3.37e-001  -4.0 1.38e-002    -
 7.55e-001 3.52e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  30 1.7892547e+000 0.00e+000 1.84e-001  -4.2 1.80e-002    -
 9.09e-001 3.75e-001f  1
  31 1.7365399e+000 0.00e+000 2.91e-001  -4.6 1.57e-002    -
 8.49e-001 2.52e-001f  1
  32 1.6895432e+000 0.00e+000 3.81e-001  -4.9 1.59e-002    -
 5.74e-001 2.18e-001f  1
```

```
  33 1.5952892e+000 0.00e+000 3.90e-001  -4.3 2.44e-002    -
 8.73e-001 3.54e-001f  1
  34 1.5497389e+000 0.00e+000 4.60e-001  -3.8 2.27e-002    -
 5.12e-001 2.81e-001f  1
  35 1.5251611e+000 0.00e+000 3.17e-001  -6.1 1.38e-002    -
 4.15e-001 1.55e-001f  1
  36 1.4604826e+000 0.00e+000 1.77e-001  -4.0 2.29e-002    -
 5.03e-001 3.36e-001f  1
  37 1.3963247e+000 0.00e+000 1.91e-001  -3.8 1.68e-002    -
 6.90e-001 5.85e-001f  1
  38 1.3796575e+000 0.00e+000 2.32e-001  -4.2 8.29e-003    -
 8.16e-001 3.01e-001f  1
  39 1.3566979e+000 0.00e+000 2.37e-001 -10.1 1.11e-002    -
 4.98e-001 2.31e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  40 1.3029610e+000 0.00e+000 9.05e-002  -5.2 1.72e-002    -
 7.45e-001 3.85e-001f  1
  41 1.2783512e+000 0.00e+000 3.01e-001  -5.4 1.57e-002    -  1.00e
+000 2.05e-001f  1
  42 1.2370704e+000 0.00e+000 2.40e-001  -3.9 1.19e-002    -
 6.79e-001 5.93e-001f  1
  43 1.2092238e+000 0.00e+000 1.04e-001  -4.1 8.64e-003    -
 5.07e-001 3.74e-001f  1
  44 1.1877529e+000 0.00e+000 8.27e-002  -4.1 8.41e-003    -
 4.89e-001 4.19e-001f  1
  45 1.1507425e+000 0.00e+000 1.89e-001  -4.3 1.14e-002    -  1.00e
+000 5.17e-001f  1
  46 1.1379104e+000 0.00e+000 2.90e-001  -4.9 1.32e-002    -
 6.55e-001 1.40e-001f  1
  47 1.1441468e+000 0.00e+000 1.07e-001  -3.5 1.69e-002    -
 3.60e-001 7.40e-001f  1
  48 1.1103969e+000 0.00e+000 2.88e-001  -4.0 6.66e-003    -  1.00e
+000 7.26e-001f  1
  49 1.1129293e+000 0.00e+000 1.41e-001  -3.7 3.23e-003    -  1.00e
+000 1.00e+000f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  50 1.0856600e+000 0.00e+000 6.85e-002  -3.8 5.30e-003    -
 8.63e-001 1.00e+000f  1
  51 1.0604318e+000 0.00e+000 6.74e-002  -3.8 8.47e-003    -
 9.91e-001 8.93e-001f  1
  52 1.0377758e+000 0.00e+000 3.46e-001  -4.2 7.24e-003    -  1.00e
+000 4.14e-001f  1
  53 1.0157480e+000 0.00e+000 4.10e-001 -10.2 1.23e-002    -
 7.03e-001 2.74e-001f  1
  54 9.8883479e-001 0.00e+000 2.68e-001  -5.7 1.52e-002    -  1.00e
+000 3.33e-001f  1
  55 9.7577103e-001 0.00e+000 2.44e-001  -5.6 1.50e-002    -
 9.05e-001 1.78e-001f  1
  56 9.5582931e-001 0.00e+000 4.26e-001  -5.6 1.69e-002    -
 7.39e-001 2.68e-001f  1
  57 9.4519785e-001 0.00e+000 2.35e-001  -5.7 1.76e-002    -
 8.17e-001 1.43e-001f  1
```

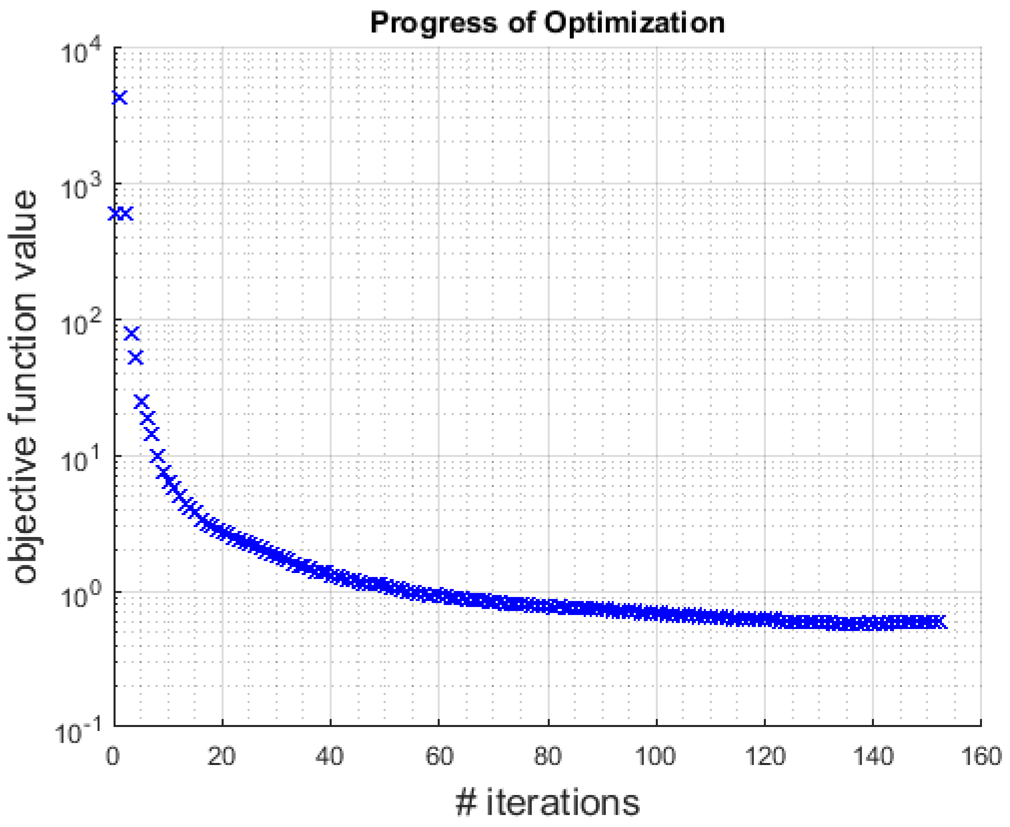
```
  58 9.3160781e-001 0.00e+000 2.55e-001  -4.7 1.35e-002    -
 4.87e-001 2.47e-001f  1
  59 9.5550662e-001 0.00e+000 1.97e-001  -3.9 2.64e-002    -
 5.46e-001 1.00e+000f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  60 9.2389102e-001 0.00e+000 8.65e-002  -4.2 1.01e-002    -
 7.20e-001 5.90e-001f  1
  61 9.0676276e-001 0.00e+000 1.51e-001  -4.3 6.01e-003    -
 9.90e-001 6.08e-001f  1
  62 9.0048428e-001 0.00e+000 1.04e-001  -4.3 4.52e-003    -
 7.32e-001 3.61e-001f  1
  63 8.8782510e-001 0.00e+000 4.46e-001  -4.5 6.80e-003    -
 9.72e-001 5.29e-001f  1
  64 8.8184512e-001 0.00e+000 2.94e-001  -4.7 8.54e-003    -
 9.28e-001 1.99e-001f  1
  65 8.7020865e-001 0.00e+000 3.57e-001  -5.3 1.32e-002    -
 8.65e-001 2.31e-001f  1
  66 8.5737707e-001 0.00e+000 3.96e-001  -5.9 1.51e-002    -
 9.05e-001 2.22e-001f  1
  67 8.4561320e-001 0.00e+000 2.11e-001  -5.7 1.71e-002    -
 5.74e-001 1.87e-001f  1
  68 8.4901115e-001 0.00e+000 4.49e-001  -4.1 2.34e-002    -
 8.98e-001 1.00e+000f  1
  69 8.3361733e-001 0.00e+000 1.15e-001  -4.3 3.76e-003    -
 9.33e-001 7.87e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  70 8.2498469e-001 0.00e+000 9.42e-002  -4.3 4.80e-003    -
 7.77e-001 1.00e+000f  1
  71 8.2109889e-001 0.00e+000 2.69e-001  -4.9 6.59e-003    -  1.00e
+000 1.92e-001f  1
  72 8.1122269e-001 0.00e+000 2.23e-001  -4.6 9.53e-003    -  1.00e
+000 3.64e-001f  1
  73 8.0478585e-001 0.00e+000 3.45e-001  -5.2 1.18e-002    -
 9.01e-001 1.95e-001f  1
  74 7.9436890e-001 0.00e+000 3.07e-001  -4.7 1.20e-002    -
 8.31e-001 3.24e-001f  1
  75 8.0036349e-001 0.00e+000 1.32e-001  -4.1 1.10e-002    -
 6.65e-001 1.00e+000f  1
  76 7.9072667e-001 0.00e+000 1.09e-001  -4.3 7.66e-003    -
 6.35e-001 4.24e-001f  1
  77 7.8214068e-001 0.00e+000 2.45e-001  -4.6 7.74e-003    -
 7.77e-001 3.86e-001f  1
  78 7.7555914e-001 0.00e+000 2.36e-001  -4.5 6.95e-003    -
 6.54e-001 4.21e-001f  1
  79 7.7269777e-001 0.00e+000 2.90e-001  -4.8 6.15e-003    -  1.00e
+000 1.78e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  80 7.6536031e-001 0.00e+000 1.77e-001  -5.0 9.70e-003    -
 8.26e-001 3.05e-001f  1
  81 7.6040165e-001 0.00e+000 2.38e-001  -4.9 8.65e-003    -
 8.05e-001 2.39e-001f  1
```

```
  82 7.7042260e-001 0.00e+000 3.59e-001  -4.2 2.43e-002    -
 4.61e-001 1.00e+000f  1
  83 7.5983470e-001 0.00e+000 9.36e-002  -4.5 6.64e-003    -  1.00e
+000 4.54e-001f  1
  84 7.5349853e-001 0.00e+000 8.24e-002  -4.2 2.39e-003    -  1.00e
+000 1.00e+000f  1
  85 7.5042124e-001 0.00e+000 1.06e-001  -4.3 9.42e-003    -
 8.59e-001 1.00e+000f  1
  86 7.4745863e-001 0.00e+000 2.49e-001  -4.6 9.70e-003    -  1.00e
+000 4.73e-001f  1
  87 7.4400752e-001 0.00e+000 2.39e-001  -4.9 1.66e-002    -  1.00e
+000 1.83e-001f  1
  88 7.3403506e-001 0.00e+000 1.70e-001  -5.0 2.97e-002    -
 6.55e-001 3.13e-001f  1
  89 7.3160911e-001 0.00e+000 2.04e-001  -7.0 2.18e-002    -
 5.74e-001 7.88e-002f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
  90 7.2509759e-001 0.00e+000 2.38e-001  -5.9 1.66e-002    -
 3.40e-001 2.23e-001f  1
  91 7.1743717e-001 0.00e+000 2.79e-001  -4.9 1.81e-002    -
 7.32e-001 2.78e-001f  1
  92 7.1232170e-001 0.00e+000 1.20e-001  -4.6 1.46e-002    -
 6.98e-001 2.65e-001f  1
  93 7.1050477e-001 0.00e+000 1.75e-001  -4.7 1.14e-002    -
 5.05e-001 1.34e-001f  1
  94 7.0309672e-001 0.00e+000 1.26e-001  -4.9 3.04e-002    -
 4.58e-001 2.07e-001f  1
  95 7.0073447e-001 0.00e+000 1.96e-001  -6.1 2.79e-002    -
 4.85e-001 6.85e-002f  1
  96 6.9267659e-001 0.00e+000 1.98e-001  -6.9 3.23e-002    -
 4.67e-001 2.08e-001f  1
  97 6.8501604e-001 0.00e+000 1.20e-001  -6.8 3.97e-002    -
 5.61e-001 1.68e-001f  1
  98 6.8391384e-001 0.00e+000 2.22e-001  -6.0 1.50e-002    -
 3.44e-001 6.10e-002f  1
  99 6.7888510e-001 0.00e+000 1.64e-001 -11.0 3.91e-002    -
 2.47e-001 1.05e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
 100 6.7307084e-001 0.00e+000 1.66e-001  -6.0 4.37e-002    -
 4.34e-001 9.99e-002f  1
 101 6.6777822e-001 0.00e+000 2.06e-001  -4.6 1.57e-002    -
 5.73e-001 2.74e-001f  1
 102 6.6330364e-001 0.00e+000 1.25e-001  -4.6 5.28e-003    -
 4.14e-001 6.13e-001f  1
 103 6.6056498e-001 0.00e+000 6.33e-002  -4.7 9.46e-003    -
 4.26e-001 2.09e-001f  1
 104 6.5619206e-001 0.00e+000 1.74e-001  -4.8 1.09e-002    -
 4.33e-001 2.82e-001f  1
 105 6.5210478e-001 0.00e+000 1.26e-001  -5.1 1.60e-002    -
 6.63e-001 1.86e-001f  1
 106 6.4989442e-001 0.00e+000 1.75e-001  -5.4 1.42e-002    -
 6.81e-001 1.12e-001f  1
```

```
  107 6.4503213e-001 0.00e+000 1.47e-001  -6.7 1.97e-002    -
  2.55e-001 1.85e-001f  1
  108 6.4266964e-001 0.00e+000 1.81e-001  -6.0 1.87e-002    -
  3.59e-001 9.69e-002f  1
  109 6.3779671e-001 0.00e+000 1.14e-001  -5.0 2.16e-002    -
  8.50e-001 1.91e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
  alpha_pr  ls
  110 6.3319897e-001 0.00e+000 1.37e-001  -4.7 1.08e-002    -
  4.84e-001 3.98e-001f  1
  111 6.3203262e-001 0.00e+000 8.59e-002  -4.5 7.66e-003    -
  3.71e-001 1.82e-001f  1
  112 6.3141163e-001 0.00e+000 2.20e-001 -10.7 7.46e-003    -
  3.84e-001 7.64e-002f  1
  113 6.2653912e-001 0.00e+000 6.89e-002  -5.0 2.05e-002    -
  5.58e-001 2.40e-001f  1
  114 6.2181409e-001 0.00e+000 6.72e-002  -4.9 2.32e-002    -
  3.39e-001 2.44e-001f  1
  115 6.2006604e-001 0.00e+000 1.72e-001  -4.9 7.77e-003    -
  3.82e-001 2.65e-001f  1
  116 6.1665328e-001 0.00e+000 9.09e-002  -4.9 1.71e-002    -
  4.17e-001 2.34e-001f  1
  117 6.1487729e-001 0.00e+000 1.09e-001  -4.7 1.52e-002    -
  4.54e-001 1.51e-001f  1
  118 6.1203611e-001 0.00e+000 1.23e-001  -4.8 1.14e-002    -
  3.44e-001 3.52e-001f  1
  119 6.1158737e-001 0.00e+000 1.24e-001  -6.4 8.77e-003    -
  5.07e-001 5.62e-002f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
  alpha_pr  ls
  120 6.0898066e-001 0.00e+000 9.64e-002  -5.8 1.81e-002    -
  5.55e-001 1.69e-001f  1
  121 6.0492436e-001 0.00e+000 1.78e-001  -5.1 1.71e-002    -
  6.54e-001 2.93e-001f  1
  122 6.0223774e-001 0.00e+000 9.77e-002  -5.8 1.68e-002    -
  3.55e-001 1.98e-001f  1
  123 6.0096853e-001 0.00e+000 6.65e-002  -4.5 8.92e-003    -
  8.01e-001 4.57e-001f  1
  124 5.9984466e-001 0.00e+000 1.37e-001  -4.6 5.77e-003    -
  5.75e-001 3.88e-001f  1
  125 5.9668600e-001 0.00e+000 1.01e-001  -5.4 7.64e-003    -
  4.66e-001 4.35e-001f  1
  126 5.9444487e-001 0.00e+000 1.68e-001  -5.2 8.51e-003    -
  4.26e-001 2.96e-001f  1
  127 5.9212229e-001 0.00e+000 7.68e-002  -5.1 1.47e-002    -
  4.83e-001 2.00e-001f  1
  128 5.9046389e-001 0.00e+000 5.84e-002  -5.4 1.62e-002    -
  2.72e-001 1.25e-001f  1
  129 5.8679662e-001 0.00e+000 1.68e-001  -4.8 1.15e-002    -
  5.48e-001 4.53e-001f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
  alpha_pr  ls
  130 5.8482199e-001 0.00e+000 1.38e-001  -4.7 8.52e-003    -
  6.12e-001 6.15e-001f  1
```

```
 131 5.8451850e-001 0.00e+000 1.56e-001  -5.8 5.82e-003    -
 7.41e-001 7.12e-002f  1
 132 5.8170999e-001 0.00e+000 1.04e-001  -4.9 1.07e-002    -
 4.45e-001 5.00e-001f  1
 133 5.8047863e-001 0.00e+000 7.41e-002  -5.0 7.53e-003    -
 6.71e-001 2.44e-001f  1
 134 5.7827142e-001 0.00e+000 1.10e-001  -5.1 4.70e-003    -
 4.44e-001 7.79e-001f  1
 135 5.7705585e-001 0.00e+000 5.69e-002  -4.8 6.62e-003    -
 5.45e-001 6.06e-001f  1
 136 5.7427374e-001 0.00e+000 4.56e-002  -5.1 9.56e-003    -
 5.44e-001 6.18e-001f  1
 137 5.7314281e-001 0.00e+000 5.03e-002  -5.3 3.73e-003    -
 5.97e-001 4.12e-001f  1
 138 5.7016773e-001 0.00e+000 4.15e-002  -5.8 8.85e-003    -
 3.57e-001 4.20e-001f  1
 139 5.8442083e-001 0.00e+000 1.68e-001  -4.3 1.58e-002    -
 9.93e-001 5.00e-001f  2
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
 140 5.7863033e-001 0.00e+000 4.69e-002  -4.3 1.27e-003    -  1.00e
+000 1.00e+000f  1
 141 5.7843834e-001 0.00e+000 2.27e-002  -4.3 2.49e-003    -  1.00e
+000 1.00e+000f  1
 142 5.8025454e-001 0.00e+000 2.66e-002  -4.3 3.60e-003    -  1.00e
+000 1.00e+000f  1
 143 5.8251721e-001 0.00e+000 2.36e-002  -4.3 3.29e-003    -  1.00e
+000 1.00e+000f  1
 144 5.8404053e-001 0.00e+000 1.16e-001  -4.3 2.25e-003    -  1.00e
+000 1.00e+000f  1
 145 5.8756605e-001 0.00e+000 2.54e-002  -4.3 2.61e-003    -
 6.47e-001 1.00e+000f  1
 146 5.8730472e-001 0.00e+000 2.95e-002  -4.3 2.41e-003    -  1.00e
+000 1.00e+000f  1
 147 5.8778169e-001 0.00e+000 1.80e-002  -4.3 2.27e-003    -
 9.17e-001 1.00e+000f  1
 148 5.8895487e-001 0.00e+000 2.21e-002  -4.3 3.17e-003    -  1.00e
+000 1.00e+000f  1
 149 5.9032942e-001 0.00e+000 1.87e-002  -4.3 3.14e-003    -  1.00e
+000 1.00e+000f  1
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
 alpha_pr  ls
 150 5.8979331e-001 0.00e+000 1.42e-002  -4.3 1.40e-003    -  1.00e
+000 1.00e+000f  1
 151 5.8997398e-001 0.00e+000 1.74e-002  -4.3 3.08e-003    -  1.00e
+000 1.00e+000f  1
 152 5.9062996e-001 0.00e+000 1.82e-002  -4.3 3.03e-003    -  1.00e
+000 1.00e+000f  1

Number of Iterations....: 152

                                   (scaled)                 (unscaled)
Objective...............:  5.9062996324487893e-001
 5.9062996324487893e-001
```

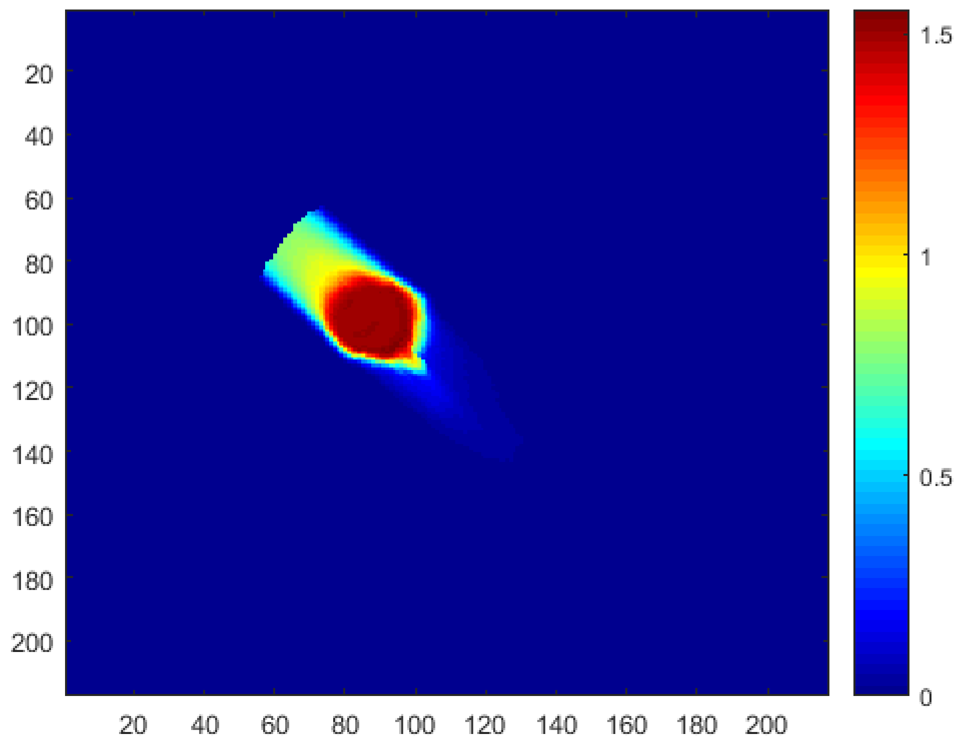
```
Dual infeasibility......:  1.8235602127227444e-002
 1.8235602127227444e-002
Constraint violation....:  0.0000000000000000e+000
 0.0000000000000000e+000
Complementarity.........:  4.5121205258897549e-005
 4.5121205258897549e-005
Overall NLP error.......:  1.8235602127227444e-002
 1.8235602127227444e-002


Number of objective function evaluations             = 164
Number of objective gradient evaluations             = 153
Number of equality constraint evaluations            = 0
Number of inequality constraint evaluations          = 0
Number of equality constraint Jacobian evaluations   = 0
Number of inequality constraint Jacobian evaluations = 0
Number of Lagrangian Hessian evaluations             = 0
Total CPU secs in IPOPT (w/o function evaluations)   =      4.837
Total CPU secs in NLP function evaluations           =    111.572

EXIT: Solved To Acceptable Level.
Calculating final cubes...
```



Plot the Resulting Dose Slice

Let's plot the transversal iso-center dose slice

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



Change Radiosensitivity

The previous treatment plan was optimized using an photon alpha-beta ratio of 2. Now, Let's change the radiosensitivity by adapting alphaX. This will change the photon alpha-beta ratio from 2 to 10.

```
for i = 1:size(cst,1)
    cst{i,5}.alphaX      = 0.5;
    cst{i,5}.TissueClass = 2;
end
```

Recalculate Plan

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

```
resultGUI_tissue = matRad_calcDoseDirect(ct,stf,pln,cst,resultGUI.w);
```

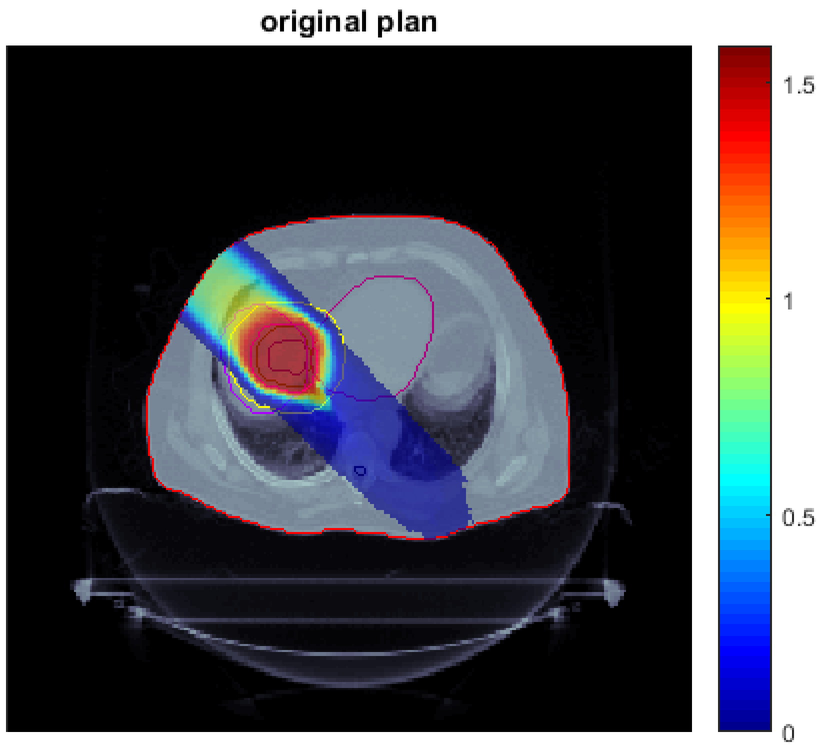
```
matRad: loading biological base data... done.
matRad: Particle dose calculation...
Beam 1 of 1:
matRad: calculate radiological depth cube...done.
matRad: calculate lateral cutoff...done.
```

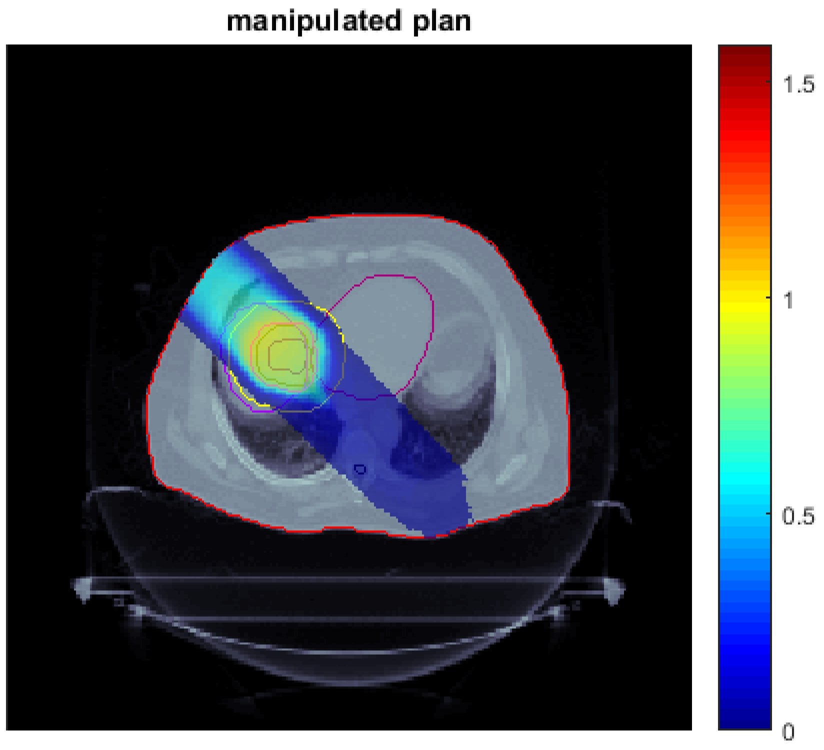
```
Progress: 100.00 %
```

Result Comparison

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

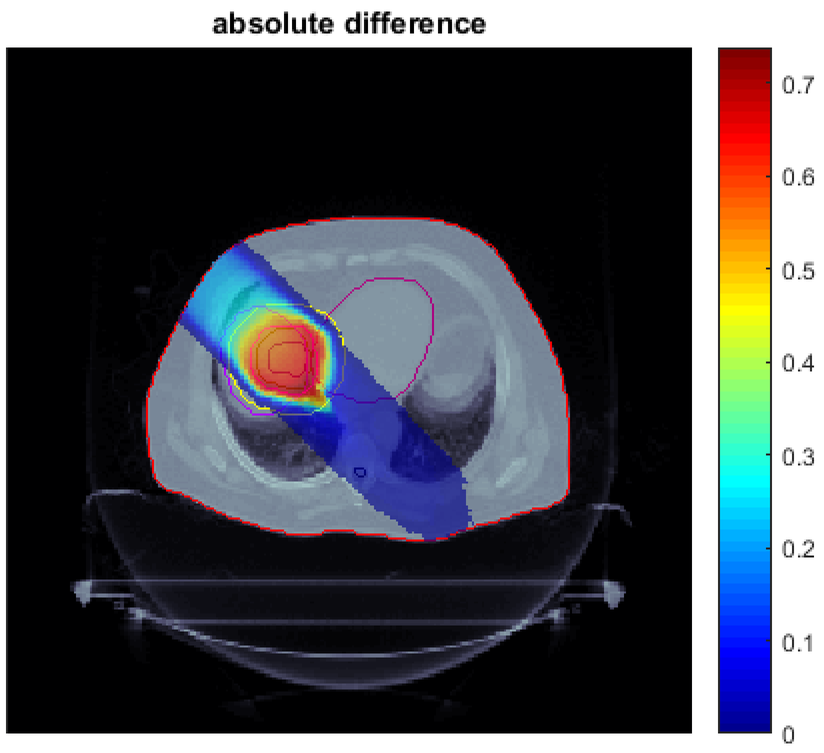
```
plane = 3;
doseWindow = [0 max([resultGUI.RBExDose(:);
 resultGUI_tissue.RBExDose(:)])];

figure,title('original plan')
[~,~,~,~,~] =
 matRad_plotSliceWrapper(gca,ct,cst,1,resultGUI.RBExDose,plane,slice,
[],[],colorcube,[],doseWindow,[]);
figure,title('manipulated plan')
[~,~,~,~,~] =
 matRad_plotSliceWrapper(gca,ct,cst,1,resultGUI_tissue.RBExDose,plane,slice,
[],[],colorcube,[],doseWindow,[]);
```





At this point we would like to see the absolute difference of the original optimization and the recalculation.

```
absDiffCube = resultGUI.RBExDose-resultGUI_tissue.RBExDose;
figure,title('absolute difference')
[~,~,~,~,~] =
 matRad_plotSliceWrapper(gca,ct,cst,1,absDiffCube,plane,slice,[],
[],colorcube);
```



Published with MATLAB® R2017a