
Example: Photon Treatment Plan

Table of Contents

.....	1
Patient Data Import	1
Treatment Plan	3
Generate Beam Geometry STF	4
Dose Calculation	5
Inverse Optimization for IMRT	6
Plot the Resulting Dose Slice	13
Now let's create another treatment plan but this time use a coarser beam spacing.	14
Visual Comparison of results	23
Obtain dose statistics	26

%%%

Copyright 2017 the matRad development team.

This file is part of the matRad project. It is subject to the license terms in the LICENSE file found in the top-level directory of this distribution and at <https://github.com/e0404/matRad/LICENSES.txt>. No part of the matRad project, including this file, may be copied, modified, propagated, or distributed except according to the terms contained in the LICENSE file.

%%%

In this example we will show (i) how to load patient data into matRad (ii) how to setup a photon dose calculation and (iii) how to inversely optimize beamlet intensities (iv) how to visually and quantitatively evaluate the result

Patient Data Import

Let's begin with a clear Matlab environment. First, import the TG119 phantom 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('TG119.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: {[167x167x129 double]}
 resolution: [1x1 struct]
```

```
cubeDim: [167 167 129]
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, 3 structures are defined in the cst

```
cst
```

```
cst =
```

```
3x6 cell array
```

```
Columns 1 through 5
```

[0]	'BODY'	'OAR'	{1x1 cell}	[1x1 struct]
[1]	'Core'	'OAR'	{1x1 cell}	[1x1 struct]
[2]	'OuterTarget'	'TARGET'	{1x1 cell}	[1x1 struct]

```
Column 6
```

```
[1x1 struct]
[1x1 struct]
[1x1 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-radiosensitivity parameter of the linear quadratic model. These parameter are required for biological treatment planning using a variable RBE. Let's output the meta optimization parameter of the target structure:

```
ixTarget = 3;
cst{ixTarget,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 making it 'expensive/costly' for the optimizer to over and underdose the target structure (both are equally important).

```
cst{ixTarget,6}

ans =

    struct with fields:

        type: 'square deviation'
        penalty: 1000
        dose: 50
        EUD: NaN
        volume: NaN
        robustness: 'none'
```

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 case we want to use photons. 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 'Generic'. By this means matRad will look for 'photons_Generic.mat' in our root directory and will use the data provided in there for dose calculation

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

Define the flavor of biological optimization 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 are using photons, we simply set the parameter to 'none' thereby indicating the physical dose should be optimized.

```
pln.bioOptimization = 'none';
```

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. In this case, we define linear spaced beams from 0 degree to 359 degree in 30 degree steps. This results in 12 beams. All corresponding couch angles are set to 0 at this point. Moreover, we set the bixelWidth to 5, which results in a beamlet size of 5 x 5 mm in the isocenter plane. The number of fractions is set to 30. Be advised that matRad is always optimizing the fraction dose.

```
pln.gantryAngles    = [0:30:359];
pln.couchAngles     = zeros(1,numel(pln.gantryAngles));
pln.bixelWidth      = 5;
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);
```

Enable sequencing and disable direct aperture optimization (DAO) for now. A DAO optimization is shown in a separate example. The multileaf collimator leaf sequencing algorithm stratifies each beam in N static shape segments.

```
pln.runSequencing = 1;
pln.runDAO        = 0;
```

and et voila our treatment plan is ready. Lets have a look at it:

```
pln

pln =

    struct with fields:

        radiationMode: 'photons'
        machine: 'Generic'
        bioOptimization: 'none'
        gantryAngles: [0 30 60 90 120 150 180 210 240 270 300 330]
        couchAngles: [0 0 0 0 0 0 0 0 0 0 0 0]
        bixelWidth: 5
        numOfFractions: 30
        numOfBeams: 12
        numOfVoxels: 3597681
        voxelDimensions: [167 167 129]
        isoCenter: [12x3 double]
        runSequencing: 1
        runDAO: 0
```

Generate Beam Geometry STF

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

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

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

Lets display the beam geometry information of the 6th beam

```
stf(6)
```

```
ans =
```

```
    struct with fields:

        gantryAngle: 150
        couchAngle: 0
        bixelWidth: 5
        radiationMode: 'photons'
        SAD: 1000
        isoCenter: [250.7385 236.3393 162.6736]
        numOfRays: 304
```

```

ray: [1×304 struct]
sourcePoint_bev: [0 -1000 0]
sourcePoint: [500.0000 866.0254 0]
numOfBixelsPerRay: [1×304 double]
totalNumOfBixels: 304

```

Dose Calculation

Lets generate dosimetric information by pre-computing dose influence matrices for unit beamlet intensities. Having dose influences available allows then later on inverse optimization.

```

dij      = matRad_calcPhotonDose(ct,stf,pln,cst);

matRad: Photon dose calculation...
Beam 1 of 12:
matRad: calculate radiological depth cube...done
      SSD = 939mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 939 mm ...
Progress: 100.00 %
Beam 2 of 12:
matRad: calculate radiological depth cube...done
      SSD = 930mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 930 mm ...
Progress: 100.00 %
Beam 3 of 12:
matRad: calculate radiological depth cube...done
      SSD = 884mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 884 mm ...
Progress: 100.00 %
Beam 4 of 12:
matRad: calculate radiological depth cube...done
      SSD = 850mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 850 mm ...
Progress: 100.00 %
Beam 5 of 12:
matRad: calculate radiological depth cube...done
      SSD = 827mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 827 mm ...
Progress: 100.00 %
Beam 6 of 12:
matRad: calculate radiological depth cube...done
      SSD = 894mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 894 mm ...
Progress: 100.00 %
Beam 7 of 12:
matRad: calculate radiological depth cube...done
      SSD = 908mm

```

```

matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 908 mm ...
Progress: 100.00 %
Beam 8 of 12:
matRad: calculate radiological depth cube...done
SSD = 897mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 897 mm ...
Progress: 100.00 %
Beam 9 of 12:
matRad: calculate radiological depth cube...done
SSD = 827mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 827 mm ...
Progress: 100.00 %
Beam 10 of 12:
matRad: calculate radiological depth cube...done
SSD = 850mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 850 mm ...
Progress: 100.00 %
Beam 11 of 12:
matRad: calculate radiological depth cube...done
SSD = 878mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 878 mm ...
Progress: 100.00 %
Beam 12 of 12:
matRad: calculate radiological depth cube...done
SSD = 930mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 930 mm ...
Progress: 100.00 %

```

Inverse Optimization for IMRT

The goal of the fluence optimization is to find a set of beamlet/pencil beam weights which yield the best possible dose distribution according to the clinical objectives and constraints underlying the radiation treatment. Once the optimization has finished, trigger once the GUI to visualize the optimized dose cubes.

```

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

```

```

*****
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.....:      3324
      variables with only lower bounds:      3324
      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 1.7261147e+002 0.00e+000 5.54e+000   0.0 0.00e+000   - 0.00e
+000 0.00e+000   0
   1 2.4690415e+002 0.00e+000 5.94e+000  -0.4 6.80e-001   -
9.51e-001 2.78e-001f 1
   2 2.5570216e+001 0.00e+000 3.80e+000  -0.8 9.10e-002   - 1.00e
+000 1.00e+000f 1
   3 1.7599984e+001 0.00e+000 7.45e-001  -1.7 1.89e-002   -
9.96e-001 1.00e+000f 1
   4 7.6233696e+000 0.00e+000 6.41e-001  -2.5 4.12e-002   - 1.00e
+000 1.00e+000f 1
   5 4.2270947e+000 0.00e+000 3.86e-001  -3.5 3.17e-002   - 1.00e
+000 1.00e+000f 1
   6 2.8928632e+000 0.00e+000 4.54e-001  -4.1 4.99e-002   - 1.00e
+000 7.83e-001f 1
   7 2.3212797e+000 0.00e+000 1.42e-001  -4.4 2.63e-002   - 1.00e
+000 6.90e-001f 1
   8 2.2392089e+000 0.00e+000 9.42e-002  -5.7 1.00e-002   - 1.00e
+000 5.35e-001f 1
   9 2.1682098e+000 0.00e+000 1.28e-001  -6.8 1.98e-002   - 1.00e
+000 2.46e-001f 1
iter   objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
alpha_pr  ls
  10 2.1012807e+000 0.00e+000 1.73e-001  -7.6 2.97e-002   - 1.00e
+000 1.38e-001f 1
  11 2.0149822e+000 0.00e+000 2.78e-001  -8.0 3.94e-002   - 1.00e
+000 1.34e-001f 1
  12 1.9515189e+000 0.00e+000 2.92e-001  -7.9 4.19e-002   - 1.00e
+000 9.92e-002f 1
  13 1.9000904e+000 0.00e+000 3.62e-001  -7.8 3.42e-002   - 1.00e
+000 9.61e-002f 1
  14 1.8026719e+000 0.00e+000 4.08e-001  -7.7 3.70e-002   - 1.00e
+000 1.87e-001f 1
  15 1.7528795e+000 0.00e+000 2.31e-001  -7.8 4.92e-002   - 1.00e
+000 8.25e-002f 1
  16 1.7322128e+000 0.00e+000 5.39e-001  -7.9 4.32e-002   - 1.00e
+000 4.01e-002f 1
  17 1.6841204e+000 0.00e+000 3.67e-001  -4.4 4.36e-002   - 1.00e
+000 8.54e-002f 1

```

```

18 1.6326554e+000 0.00e+000 3.28e-001 -3.7 2.41e-002 - 1.00e
+000 1.70e-001f 1
19 1.6381464e+000 0.00e+000 8.23e-001 -3.2 1.10e-001 -
8.79e-001 1.00e+000f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
20 1.5953433e+000 0.00e+000 1.89e-001 -3.3 8.76e-003 - 1.00e
+000 1.00e+000f 1
21 1.5261751e+000 0.00e+000 1.30e-001 -4.8 4.08e-002 - 1.00e
+000 8.06e-001f 1
22 1.4991229e+000 0.00e+000 1.93e-001 -6.4 3.49e-002 - 1.00e
+000 3.89e-001f 1
23 1.4483759e+000 0.00e+000 7.43e-002 -4.1 5.21e-002 -
9.99e-001 4.18e-001f 1
24 1.4156142e+000 0.00e+000 1.10e-001 -5.1 3.21e-002 - 1.00e
+000 3.21e-001f 1
25 1.3903776e+000 0.00e+000 1.16e-001 -4.7 3.20e-002 - 1.00e
+000 2.16e-001f 1
26 1.3609081e+000 0.00e+000 1.07e-001 -5.4 3.37e-002 - 1.00e
+000 2.69e-001f 1
27 1.3375518e+000 0.00e+000 1.29e-001 -11.0 2.92e-002 -
7.73e-001 2.30e-001f 1
28 1.3228178e+000 0.00e+000 1.34e-001 -5.7 2.50e-002 -
7.54e-001 1.67e-001f 1
29 1.3042292e+000 0.00e+000 9.62e-002 -5.4 3.04e-002 - 1.00e
+000 1.73e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
30 1.2832818e+000 0.00e+000 1.37e-001 -6.2 3.35e-002 -
9.69e-001 2.03e-001f 1
31 1.2704239e+000 0.00e+000 1.68e-001 -7.8 3.40e-002 -
8.40e-001 1.34e-001f 1
32 1.2544862e+000 0.00e+000 2.16e-001 -6.6 3.28e-002 - 1.00e
+000 2.03e-001f 1
33 1.2477354e+000 0.00e+000 6.98e-002 -5.6 3.70e-002 -
8.97e-001 8.21e-002f 1
34 1.2413335e+000 0.00e+000 2.02e-001 -7.0 1.84e-002 -
5.06e-001 1.58e-001f 1
35 1.2305878e+000 0.00e+000 8.26e-002 -5.8 3.44e-002 - 1.00e
+000 1.62e-001f 1
36 1.3182100e+000 0.00e+000 2.38e-001 -3.7 1.89e-002 -
5.80e-001 1.00e+000f 1
37 1.2817816e+000 0.00e+000 8.71e-002 -3.8 1.32e-002 -
9.86e-001 7.08e-001f 1
38 1.2460896e+000 0.00e+000 1.67e-001 -3.8 9.41e-003 -
6.24e-001 1.00e+000f 1
39 1.2392897e+000 0.00e+000 1.42e-001 -3.8 5.99e-003 - 1.00e
+000 5.71e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
40 1.2143211e+000 0.00e+000 1.40e-001 -3.8 1.37e-002 -
6.26e-001 1.00e+000f 1
41 1.1928976e+000 0.00e+000 1.02e-001 -4.0 2.16e-002 -
9.69e-001 9.97e-001f 1

```


Example: Photon Treatment Plan

```

42 1.1596081e+000 0.00e+000 9.20e-002 -4.3 5.15e-003 - 1.00e
+000 1.00e+000f 1
43 1.1563170e+000 0.00e+000 6.10e-002 -4.0 3.11e-003 -
8.65e-001 1.00e+000f 1
44 1.1523144e+000 0.00e+000 2.73e-002 -3.8 1.40e-002 -
7.45e-001 1.00e+000f 1
45 1.1331420e+000 0.00e+000 4.66e-002 -4.1 1.25e-002 -
9.99e-001 8.76e-001f 1
46 1.1207105e+000 0.00e+000 1.25e-001 -4.6 2.52e-002 - 1.00e
+000 2.63e-001f 1
47 1.1568149e+000 0.00e+000 3.60e-002 -3.6 1.76e-002 -
9.32e-001 1.00e+000f 1
48 1.1251905e+000 0.00e+000 7.68e-002 -3.7 1.74e-002 - 1.00e
+000 1.00e+000f 1
49 1.1051387e+000 0.00e+000 8.45e-002 -3.7 6.93e-003 -
9.68e-001 1.00e+000f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
50 1.0649431e+000 0.00e+000 1.15e-001 -4.5 2.18e-002 - 1.00e
+000 7.40e-001f 1
51 1.0214169e+000 0.00e+000 5.22e-002 -4.7 2.53e-002 - 1.00e
+000 9.18e-001f 1
52 1.0145612e+000 0.00e+000 5.66e-002 -5.3 4.07e-002 - 1.00e
+000 9.81e-002f 1
53 1.0550545e+000 0.00e+000 8.54e-002 -3.7 1.24e-002 -
6.02e-001 1.00e+000f 1
54 1.0320485e+000 0.00e+000 7.27e-002 -3.9 2.33e-002 - 1.00e
+000 3.35e-001f 1
55 9.9691958e-001 0.00e+000 2.46e-001 -3.9 1.12e-002 - 1.00e
+000 8.85e-001f 1
56 9.8533963e-001 0.00e+000 1.99e-001 -5.6 1.09e-002 -
8.10e-001 3.25e-001f 1
57 9.6549072e-001 0.00e+000 1.28e-001 -5.7 2.05e-002 - 1.00e
+000 3.69e-001f 1
58 9.5270746e-001 0.00e+000 1.56e-001 -6.1 2.63e-002 - 1.00e
+000 2.33e-001f 1
59 9.4392188e-001 0.00e+000 1.65e-001 -6.1 3.06e-002 - 1.00e
+000 1.54e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
60 9.2896030e-001 0.00e+000 1.39e-001 -5.7 3.73e-002 -
9.76e-001 2.59e-001f 1
61 9.2317698e-001 0.00e+000 9.41e-002 -5.5 3.28e-002 -
4.01e-001 1.08e-001f 1
62 9.1609001e-001 0.00e+000 1.27e-001 -5.7 3.50e-002 -
8.09e-001 1.41e-001f 1
63 9.0587386e-001 0.00e+000 6.92e-002 -5.6 4.29e-002 -
9.29e-001 1.79e-001f 1
64 8.9841529e-001 0.00e+000 1.61e-001 -5.4 2.36e-002 -
5.44e-001 2.45e-001f 1
65 8.9038232e-001 0.00e+000 7.99e-002 -5.1 3.99e-002 - 1.00e
+000 1.79e-001f 1
66 8.8141667e-001 0.00e+000 6.17e-002 -4.7 4.70e-002 -
4.17e-001 1.78e-001f 1

```

```

67 8.7980337e-001 0.00e+000 1.50e-001 -5.4 2.55e-002 -
4.39e-001 5.84e-002f 1
68 8.7487671e-001 0.00e+000 1.13e-001 -4.7 3.64e-002 -
3.93e-001 1.39e-001f 1
69 8.6665189e-001 0.00e+000 6.31e-002 -4.6 4.33e-002 -
3.47e-001 1.88e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
70 8.6429129e-001 0.00e+000 8.92e-002 -4.5 2.81e-002 -
5.29e-001 9.38e-002f 1
71 8.5558594e-001 0.00e+000 5.72e-002 -4.9 3.32e-002 -
6.07e-001 2.37e-001f 1
72 8.4918804e-001 0.00e+000 4.64e-002 -10.8 3.40e-002 -
3.31e-001 1.54e-001f 1
73 8.4654527e-001 0.00e+000 5.14e-002 -4.6 9.39e-003 -
1.94e-001 2.35e-001f 1
74 8.4034601e-001 0.00e+000 4.40e-002 -10.8 3.00e-002 -
1.89e-001 1.76e-001f 1
75 8.3758916e-001 0.00e+000 6.30e-002 -4.9 2.34e-002 -
4.37e-001 9.63e-002f 1
76 8.3233821e-001 0.00e+000 5.77e-002 -5.6 2.14e-002 -
2.78e-001 2.10e-001f 1
77 8.2726643e-001 0.00e+000 3.61e-002 -5.1 2.47e-002 -
2.64e-001 2.02e-001f 1
78 8.3793107e-001 0.00e+000 5.54e-002 -4.1 1.29e-002 -
5.36e-001 1.00e+000f 1
79 8.3077276e-001 0.00e+000 3.98e-002 -4.4 5.33e-003 -
9.26e-001 7.94e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
80 8.2851625e-001 0.00e+000 5.17e-002 -4.4 2.07e-003 - 1.00e
+000 1.00e+000f 1
81 8.2715592e-001 0.00e+000 2.07e-002 -4.4 3.62e-003 -
8.99e-001 1.00e+000f 1
82 8.2228808e-001 0.00e+000 2.25e-002 -4.5 4.69e-003 - 1.00e
+000 1.00e+000f 1
83 8.1410503e-001 0.00e+000 2.30e-002 -5.1 7.32e-003 - 1.00e
+000 7.65e-001f 1
84 8.0880794e-001 0.00e+000 2.51e-002 -5.2 1.37e-002 - 1.00e
+000 4.65e-001f 1
85 8.0666478e-001 0.00e+000 4.73e-002 -4.6 6.31e-003 -
6.51e-001 7.06e-001f 1
86 8.0324300e-001 0.00e+000 5.22e-002 -4.8 1.25e-002 -
7.14e-001 3.84e-001f 1
87 8.0038261e-001 0.00e+000 5.34e-002 -4.9 1.24e-002 -
5.18e-001 2.97e-001f 1
88 7.9723259e-001 0.00e+000 5.53e-002 -5.2 1.70e-002 -
7.25e-001 2.39e-001f 1
89 7.9349011e-001 0.00e+000 5.03e-002 -11.0 1.97e-002 -
4.59e-001 2.56e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
90 7.9054692e-001 0.00e+000 6.87e-002 -6.2 2.23e-002 -
8.68e-001 1.98e-001f 1

```

Example: Photon Treatment Plan

```

91 7.8730530e-001 0.00e+000 5.57e-002 -5.2 1.74e-002 -
4.39e-001 3.28e-001f 1
92 7.9322592e-001 0.00e+000 1.82e-001 -4.4 3.14e-002 -
5.01e-001 1.00e+000f 1
93 7.8866756e-001 0.00e+000 2.68e-002 -4.6 1.70e-002 -
8.73e-001 5.06e-001f 1
94 7.8566156e-001 0.00e+000 5.63e-002 -4.6 5.92e-003 -
7.38e-001 1.00e+000f 1
95 7.8421618e-001 0.00e+000 8.49e-002 -5.2 1.41e-002 -
7.75e-001 1.90e-001f 1
96 7.7987146e-001 0.00e+000 4.32e-002 -5.1 1.91e-002 -
8.83e-001 3.25e-001f 1
97 7.7575353e-001 0.00e+000 3.67e-002 -5.2 2.78e-002 -
5.29e-001 2.52e-001f 1
98 7.7396226e-001 0.00e+000 6.01e-002 -6.6 1.99e-002 -
3.41e-001 1.34e-001f 1
99 7.7097100e-001 0.00e+000 5.05e-002 -5.5 2.85e-002 -
4.59e-001 1.62e-001f 1
iter   objective   inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
alpha_pr  ls
100 7.6834081e-001 0.00e+000 4.30e-002 -5.0 2.85e-002 -
3.15e-001 1.52e-001f 1
101 7.6577124e-001 0.00e+000 4.89e-002 -7.3 3.02e-002 -
2.50e-001 1.36e-001f 1
102 7.6319828e-001 0.00e+000 9.32e-002 -5.4 4.72e-002 -
8.24e-001 9.63e-002f 1
103 7.6143305e-001 0.00e+000 8.10e-002 -6.2 1.76e-002 -
9.78e-002 1.55e-001f 1
104 7.5795280e-001 0.00e+000 5.04e-002 -5.5 3.79e-002 -
4.84e-001 1.53e-001f 1
105 8.4623252e-001 0.00e+000 1.51e-001 -4.0 9.04e-002 -
4.93e-001 1.00e+000f 1
106 7.9071269e-001 0.00e+000 7.54e-002 -4.3 4.02e-002 - 1.00e
+000 5.91e-001f 1
107 7.6627662e-001 0.00e+000 1.26e-001 -4.3 1.73e-002 - 1.00e
+000 1.00e+000f 1
108 7.6481776e-001 0.00e+000 2.73e-002 -4.3 2.25e-003 - 1.00e
+000 1.00e+000f 1
109 7.6368214e-001 0.00e+000 2.83e-002 -4.3 7.67e-003 -
9.33e-001 1.00e+000f 1
iter   objective   inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
alpha_pr  ls
110 7.6338675e-001 0.00e+000 1.92e-002 -4.3 8.44e-003 - 1.00e
+000 1.00e+000f 1
111 7.6259789e-001 0.00e+000 1.78e-002 -4.3 1.03e-002 - 1.00e
+000 1.00e+000f 1
112 7.6162777e-001 0.00e+000 8.89e-003 -4.3 1.72e-003 - 1.00e
+000 1.00e+000f 1

```

Number of Iterations.....: 112

```

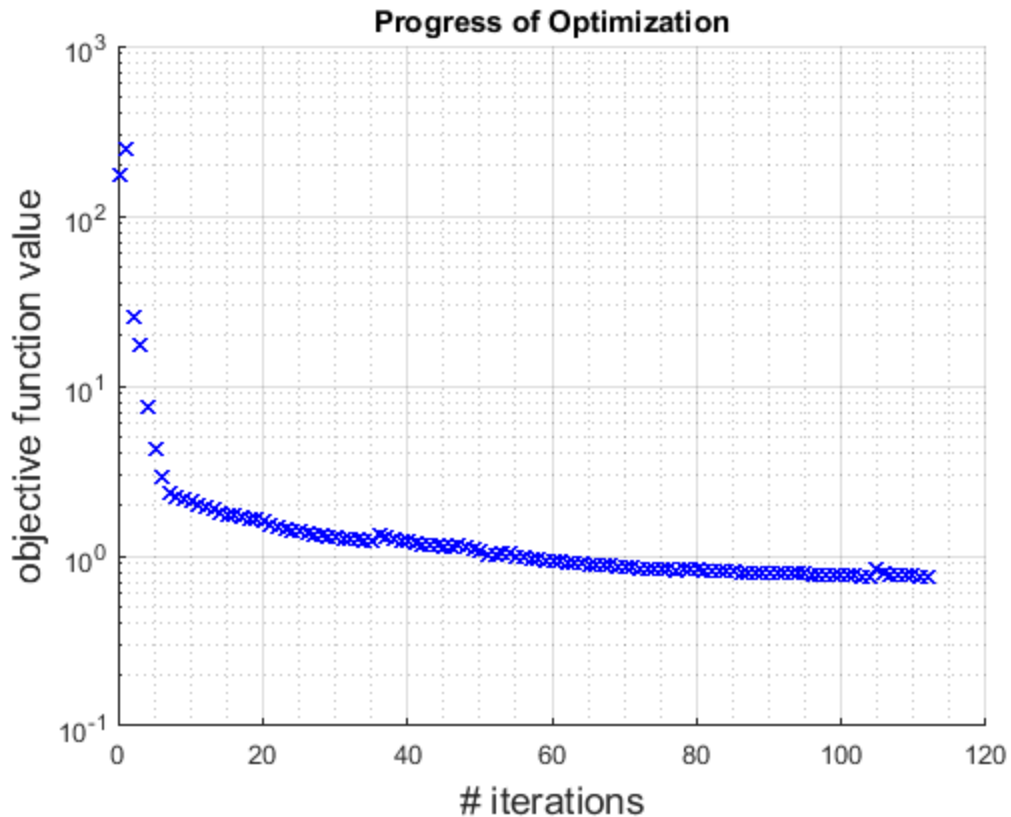
                                     (scaled)                               (unscaled)
Objective.....: 7.6162777376635393e-001
7.6162777376635393e-001

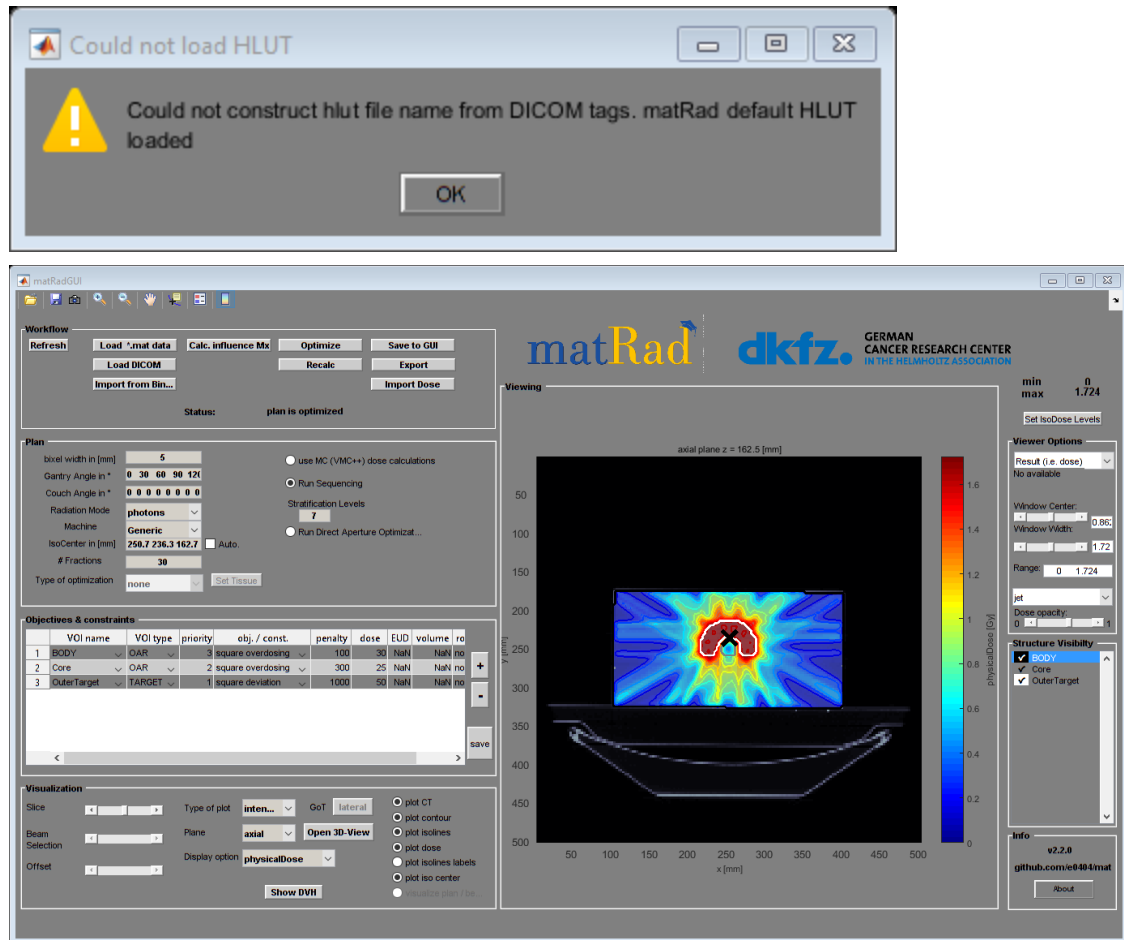
```

```
Dual infeasibility.....: 8.8915524124589442e-003
8.8915524124589442e-003
Constraint violation....: 0.0000000000000000e+000
0.0000000000000000e+000
Complementarity.....: 5.9235295952830844e-005
5.9235295952830844e-005
Overall NLP error.....: 8.8915524124589442e-003
8.8915524124589442e-003
```

```
Number of objective function evaluations      = 113
Number of objective gradient evaluations     = 113
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) = 2.629
Total CPU secs in NLP function evaluations   = 58.621
```

```
EXIT: Solved To Acceptable Level.
Calculating final cubes...
Warning: matRad default HLUT loaded
Reconversion of HU values could not be done because HLUT is not
bijective.
```

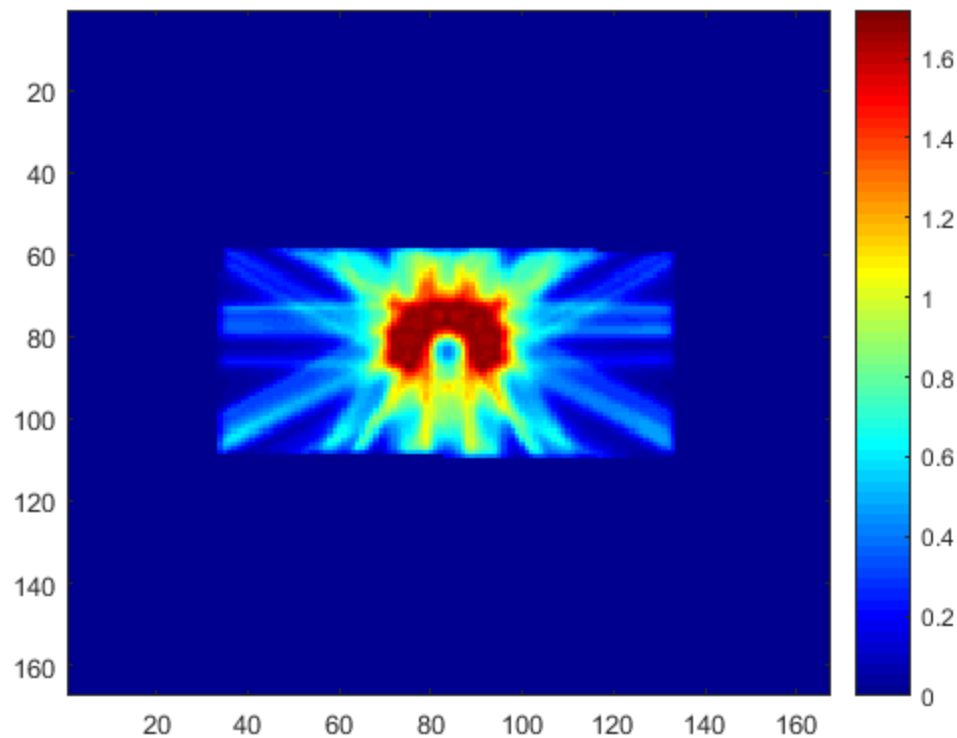




Plot the Resulting Dose Slice

Let's plot the transversal iso-center dose slice

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



Now let's create another treatment plan but this time use a coarser beam spacing.

Instead of 30 degree spacing use a 50 degree gantry beam spacing

```
pln.gantryAngles = [0:50:359];
pln.couchAngles = zeros(1,numel(pln.gantryAngles));
pln.numOfBeams = numel(pln.gantryAngles);
stf = matRad_generateStf(ct,cst,pln);
pln.isoCenter = stf.isoCenter;
dij = matRad_calcPhotonDose(ct,stf,pln,cst);
resultGUI_coarse = matRad_fluenceOptimization(dij,cst,pln);
```

```
matRad: Generating stf struct... Progress: 100.00 %
matRad: Photon dose calculation...
Beam 1 of 8:
matRad: calculate radiological depth cube...done
        SSD = 939mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 939 mm ...
Progress: 100.00 %
Beam 2 of 8:
matRad: calculate radiological depth cube...done
        SSD = 905mm
```

```

matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 905 mm ...
Progress: 100.00 %
Beam 3 of 8:
matRad: calculate radiological depth cube...done
SSD = 848mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 848 mm ...
Progress: 100.00 %
Beam 4 of 8:
matRad: calculate radiological depth cube...done
SSD = 894mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 894 mm ...
Progress: 100.00 %
Beam 5 of 8:
matRad: calculate radiological depth cube...done
SSD = 905mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 905 mm ...
Progress: 100.00 %
Beam 6 of 8:
matRad: calculate radiological depth cube...done
SSD = 840mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 840 mm ...
Progress: 100.00 %
Beam 7 of 8:
matRad: calculate radiological depth cube...done
SSD = 878mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 878 mm ...
Progress: 100.00 %
Beam 8 of 8:
matRad: calculate radiological depth cube...done
SSD = 938mm
matRad: Uniform primary photon fluence -> pre-compute kernel
convolution for SSD = 938 mm ...
Progress: 100.00 %

```

```

*****
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.....:      2207
      variables with only lower bounds:      2207
      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 1.6835152e+002 0.00e+000 5.54e+000   0.0 0.00e+000   - 0.00e
+000 0.00e+000 0
  1 9.0229894e+001 0.00e+000 5.64e+000  -0.3 1.03e+000   -
9.77e-001 2.67e-001f 1
  2 7.2225245e+001 0.00e+000 4.15e+000  -6.6 1.45e-001   - 1.00e
+000 1.00e+000f 1
  3 1.5886846e+001 0.00e+000 2.49e+000  -1.6 6.87e-002   -
9.99e-001 1.00e+000f 1
  4 1.2711254e+001 0.00e+000 6.60e-001  -1.7 2.21e-002   -
9.28e-001 1.00e+000f 1
  5 7.5346485e+000 0.00e+000 4.25e-001  -2.0 5.31e-002   -
9.98e-001 1.00e+000f 1
  6 4.3820387e+000 0.00e+000 3.54e-001  -2.8 6.10e-002   - 1.00e
+000 1.00e+000f 1
  7 3.2914364e+000 0.00e+000 5.76e-001  -3.4 8.20e-002   - 1.00e
+000 1.00e+000f 1
  8 2.3991188e+000 0.00e+000 1.84e-001  -3.8 2.75e-002   - 1.00e
+000 8.41e-001f 1
  9 2.2613266e+000 0.00e+000 1.46e-001  -4.8 9.15e-003   - 1.00e
+000 1.00e+000f 1
iter   objective   inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
alpha_pr  ls
 10 2.1628239e+000 0.00e+000 1.46e-001  -5.8 2.27e-002   - 1.00e
+000 3.33e-001f 1
 11 2.0423020e+000 0.00e+000 1.37e-001  -4.6 4.60e-002   - 1.00e
+000 2.25e-001f 1
 12 1.9278596e+000 0.00e+000 5.51e-001  -4.0 5.47e-002   - 1.00e
+000 1.98e-001f 1
 13 1.8476477e+000 0.00e+000 8.75e-001  -2.9 1.32e-001   -
6.38e-001 1.00e+000f 1
 14 1.7538306e+000 0.00e+000 1.58e-001  -4.2 2.31e-002   -
9.90e-001 1.00e+000f 1
 15 1.6968139e+000 0.00e+000 7.91e-002  -4.4 3.37e-002   - 1.00e
+000 1.00e+000f 1
 16 1.6593228e+000 0.00e+000 5.78e-002  -5.7 3.13e-002   - 1.00e
+000 6.94e-001f 1
 17 1.6351462e+000 0.00e+000 1.14e-001  -6.2 2.83e-002   - 1.00e
+000 2.83e-001f 1
 18 1.6021937e+000 0.00e+000 9.55e-002  -7.1 4.55e-002   - 1.00e
+000 2.10e-001f 1
 19 1.5710715e+000 0.00e+000 5.16e-002  -7.4 3.70e-002   - 1.00e
+000 1.75e-001f 1

```


iter	objective	inf_pr	inf_du	lg(mu)	d	lg(rg)	alpha_du
alpha_pr	ls						
20	1.5499955e+000	0.00e+000	2.04e-001	-4.2	5.32e-002	-	
9.31e-001	9.40e-002f	1					
21	1.4927035e+000	0.00e+000	4.58e-001	-3.7	9.24e-002	-	
8.51e-001	3.20e-001f	1					
22	1.5233347e+000	0.00e+000	4.25e-001	-3.3	7.05e-002	-	
8.15e-001	1.00e+000f	1					
23	1.4751530e+000	0.00e+000	1.02e-001	-3.3	1.23e-002	-	1.00e
+000	1.00e+000f	1					
24	1.4515605e+000	0.00e+000	9.01e-002	-4.0	1.94e-002	-	
9.99e-001	1.00e+000f	1					
25	1.4320297e+000	0.00e+000	1.03e-001	-5.3	2.17e-002	-	1.00e
+000	6.07e-001f	1					
26	1.4112400e+000	0.00e+000	6.73e-002	-5.8	2.85e-002	-	1.00e
+000	3.49e-001f	1					
27	1.4043540e+000	0.00e+000	1.38e-001	-5.2	1.59e-002	-	
7.77e-001	1.71e-001f	1					
28	1.3872250e+000	0.00e+000	5.48e-002	-5.6	3.60e-002	-	1.00e
+000	2.05e-001f	1					
29	1.3571939e+000	0.00e+000	9.36e-002	-5.2	5.12e-002	-	1.00e
+000	3.43e-001f	1					
iter	objective	inf_pr	inf_du	lg(mu)	d	lg(rg)	alpha_du
alpha_pr	ls						
30	1.3490060e+000	0.00e+000	7.56e-002	-4.4	3.66e-002	-	1.00e
+000	1.51e-001f	1					
31	1.3307865e+000	0.00e+000	6.89e-002	-4.0	3.94e-002	-	
7.81e-001	4.97e-001f	1					
32	1.3193885e+000	0.00e+000	2.22e-001	-3.8	1.21e-002	-	
8.77e-001	1.00e+000f	1					
33	1.3126632e+000	0.00e+000	1.27e-001	-9.8	1.36e-002	-	
7.87e-001	3.78e-001f	1					
34	1.2993745e+000	0.00e+000	1.22e-001	-4.7	2.45e-002	-	
7.16e-001	4.56e-001f	1					
35	1.2922290e+000	0.00e+000	1.23e-001	-5.4	2.75e-002	-	
7.82e-001	2.03e-001f	1					
36	1.2803522e+000	0.00e+000	7.92e-002	-11.0	4.21e-002	-	
3.33e-001	2.09e-001f	1					
37	1.2674119e+000	0.00e+000	9.50e-002	-5.5	6.16e-002	-	
8.71e-001	1.64e-001f	1					
38	1.2622395e+000	0.00e+000	9.00e-002	-5.1	3.41e-002	-	
6.76e-001	1.17e-001f	1					
39	1.2463301e+000	0.00e+000	7.60e-002	-4.9	4.27e-002	-	
7.32e-001	2.61e-001f	1					
iter	objective	inf_pr	inf_du	lg(mu)	d	lg(rg)	alpha_du
alpha_pr	ls						
40	1.2400471e+000	0.00e+000	1.31e-001	-4.3	1.40e-002	-	
6.61e-001	2.50e-001f	1					
41	1.2325859e+000	0.00e+000	1.16e-001	-4.2	1.88e-002	-	
4.38e-001	2.61e-001f	1					
42	1.2276620e+000	0.00e+000	1.02e-001	-10.3	2.19e-002	-	
5.80e-001	1.47e-001f	1					
43	1.2139467e+000	0.00e+000	1.05e-001	-5.2	3.63e-002	-	1.00e
+000	2.69e-001f	1					

Example: Photon Treatment Plan

```

44 1.2092307e+000 0.00e+000 2.04e-001 -5.7 2.76e-002 -
7.76e-001 1.18e-001f 1
45 1.2025930e+000 0.00e+000 1.27e-001 -6.7 2.58e-002 -
6.64e-001 1.64e-001f 1
46 1.1918803e+000 0.00e+000 1.49e-001 -5.4 3.25e-002 - 1.00e
+000 2.49e-001f 1
47 1.1811352e+000 0.00e+000 3.80e-002 -4.5 3.64e-002 - 1.00e
+000 2.92e-001f 1
48 1.1765723e+000 0.00e+000 8.13e-002 -4.8 2.33e-002 -
8.55e-001 2.03e-001f 1
49 1.1704586e+000 0.00e+000 6.06e-002 -5.4 3.19e-002 - 1.00e
+000 2.01e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
50 1.1597742e+000 0.00e+000 2.92e-002 -4.6 3.54e-002 -
4.98e-001 4.10e-001f 1
51 1.1565150e+000 0.00e+000 1.99e-001 -4.2 1.10e-002 - 1.00e
+000 4.55e-001f 1
52 1.1532504e+000 0.00e+000 8.96e-002 -6.3 1.78e-002 -
5.69e-001 1.64e-001f 1
53 1.1398902e+000 0.00e+000 4.16e-002 -4.8 3.75e-002 -
5.61e-001 3.97e-001f 1
54 1.1308042e+000 0.00e+000 2.79e-002 -4.3 3.85e-002 -
8.22e-001 3.60e-001f 1
55 1.1223989e+000 0.00e+000 1.25e-001 -4.7 2.04e-002 -
7.23e-001 5.16e-001f 1
56 1.2717907e+000 0.00e+000 1.59e-001 -3.4 1.47e-001 - 1.00e
+000 1.00e+000f 1
57 1.1685623e+000 0.00e+000 9.65e-002 -3.4 4.54e-002 - 1.00e
+000 1.00e+000f 1
58 1.1578186e+000 0.00e+000 7.16e-002 -3.4 1.25e-002 - 1.00e
+000 1.00e+000f 1
59 1.1514750e+000 0.00e+000 3.91e-002 -3.4 8.81e-003 - 1.00e
+000 1.00e+000f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
60 1.1495486e+000 0.00e+000 6.27e-002 -3.4 1.05e-002 - 1.00e
+000 1.00e+000f 1
61 1.1480513e+000 0.00e+000 3.68e-002 -3.4 1.09e-002 - 1.00e
+000 1.00e+000f 1
62 1.1467486e+000 0.00e+000 2.48e-002 -3.4 5.57e-003 - 1.00e
+000 1.00e+000f 1
63 1.1428323e+000 0.00e+000 4.21e-002 -3.4 1.80e-002 - 1.00e
+000 1.00e+000f 1
64 1.1390424e+000 0.00e+000 3.00e-002 -3.4 8.28e-003 - 1.00e
+000 1.00e+000f 1
65 1.1357442e+000 0.00e+000 2.13e-002 -3.4 1.92e-003 - 1.00e
+000 1.00e+000f 1
66 1.1309438e+000 0.00e+000 1.76e-002 -3.4 5.23e-003 - 1.00e
+000 1.00e+000f 1
67 1.1269752e+000 0.00e+000 2.58e-002 -3.4 5.71e-003 - 1.00e
+000 1.00e+000f 1
68 1.1206528e+000 0.00e+000 3.37e-002 -3.4 1.20e-002 - 1.00e
+000 1.00e+000f 1

```

```

69 1.1120263e+000 0.00e+000 3.22e-002 -4.2 1.07e-002 - 1.00e
+000 1.00e+000f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
70 1.1017197e+000 0.00e+000 3.40e-002 -5.6 7.69e-003 - 1.00e
+000 1.00e+000f 1
71 1.0900689e+000 0.00e+000 2.89e-002 -6.2 1.87e-002 - 1.00e
+000 3.64e-001f 1
72 1.0848965e+000 0.00e+000 6.03e-002 -6.4 1.52e-002 - 1.00e
+000 1.55e-001f 1
73 1.0776577e+000 0.00e+000 6.15e-002 -4.0 4.50e-003 -
9.60e-001 8.37e-001f 1
74 1.0662697e+000 0.00e+000 1.17e-001 -4.7 1.14e-002 - 1.00e
+000 5.27e-001f 1
75 1.0570152e+000 0.00e+000 4.08e-002 -5.8 1.34e-002 - 1.00e
+000 4.43e-001f 1
76 1.0527430e+000 0.00e+000 4.43e-002 -4.1 1.14e-002 -
9.96e-001 7.81e-001f 1
77 1.0421556e+000 0.00e+000 1.25e-001 -5.0 1.09e-002 - 1.00e
+000 7.47e-001f 1
78 1.0396648e+000 0.00e+000 6.81e-002 -10.9 1.23e-002 -
8.66e-001 1.86e-001f 1
79 1.0334545e+000 0.00e+000 6.59e-002 -5.2 2.23e-002 -
7.48e-001 2.97e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
80 1.0283782e+000 0.00e+000 8.19e-002 -5.6 2.35e-002 -
5.66e-001 2.65e-001f 1
81 1.0266396e+000 0.00e+000 1.23e-001 -7.3 2.37e-002 -
8.62e-001 9.36e-002f 1
82 1.0220485e+000 0.00e+000 9.63e-002 -6.7 3.19e-002 -
9.03e-001 2.00e-001f 1
83 1.0177696e+000 0.00e+000 6.27e-002 -7.4 3.55e-002 - 1.00e
+000 1.74e-001f 1
84 1.0154361e+000 0.00e+000 1.05e-001 -5.4 2.34e-002 -
7.51e-001 1.51e-001f 1
85 1.0115662e+000 0.00e+000 4.47e-002 -4.8 3.74e-002 -
7.93e-001 1.73e-001f 1
86 1.0097005e+000 0.00e+000 7.91e-002 -7.0 2.66e-002 -
3.71e-001 1.04e-001f 1
87 1.0062117e+000 0.00e+000 7.76e-002 -4.4 4.86e-002 -
6.03e-001 1.32e-001f 1
88 1.0022194e+000 0.00e+000 6.78e-002 -4.3 3.64e-002 -
5.30e-001 1.89e-001f 1
89 9.9783179e-001 0.00e+000 8.94e-002 -10.4 3.03e-002 -
4.30e-001 2.33e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
90 9.9526408e-001 0.00e+000 1.80e-001 -5.0 1.93e-002 -
8.31e-001 2.06e-001f 1
91 9.9309042e-001 0.00e+000 1.70e-001 -5.5 2.31e-002 -
6.17e-001 1.32e-001f 1
92 9.8856582e-001 0.00e+000 8.44e-002 -4.6 2.53e-002 -
5.15e-001 2.72e-001f 1

```

```

 93 9.8498654e-001 0.00e+000 5.97e-002 -4.9 2.45e-002 -
4.22e-001 1.95e-001f 1
 94 9.8110315e-001 0.00e+000 4.79e-002 -10.8 3.71e-002 -
2.77e-001 1.37e-001f 1
 95 9.7828877e-001 0.00e+000 7.91e-002 -4.7 3.19e-002 -
6.41e-001 1.20e-001f 1
 96 9.7319549e-001 0.00e+000 7.35e-002 -4.8 4.12e-002 -
4.27e-001 1.74e-001f 1
 97 9.7125917e-001 0.00e+000 7.10e-002 -10.8 4.10e-002 -
4.44e-001 6.38e-002f 1
 98 9.6717406e-001 0.00e+000 9.47e-002 -5.8 3.02e-002 -
5.27e-001 1.91e-001f 1
 99 9.6232298e-001 0.00e+000 5.40e-002 -5.2 2.88e-002 -
6.20e-001 2.69e-001f 1
iter   objective   inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
alpha_pr  ls
100 9.6117072e-001 0.00e+000 7.34e-002 -5.2 7.63e-003 -
4.30e-001 1.95e-001f 1
101 9.5866235e-001 0.00e+000 6.79e-002 -4.4 1.38e-002 -
6.48e-001 3.40e-001f 1
102 9.5725853e-001 0.00e+000 5.68e-002 -4.6 7.50e-003 -
4.70e-001 2.90e-001f 1
103 9.5563347e-001 0.00e+000 6.05e-002 -10.6 1.12e-002 -
3.15e-001 1.72e-001f 1
104 9.5348679e-001 0.00e+000 9.81e-002 -5.0 1.77e-002 -
8.45e-001 1.64e-001f 1
105 9.4842063e-001 0.00e+000 6.63e-002 -4.6 1.90e-002 -
6.27e-001 4.98e-001f 1
106 9.4617285e-001 0.00e+000 4.40e-002 -5.3 2.20e-002 -
6.76e-001 1.85e-001f 1
107 9.9103016e-001 0.00e+000 8.35e-002 -3.8 2.16e-002 -
8.70e-001 1.00e+000f 1
108 9.7220512e-001 0.00e+000 3.09e-002 -3.9 1.50e-002 - 1.00e
+000 8.38e-001f 1
109 9.6632343e-001 0.00e+000 2.12e-001 -3.9 7.15e-003 - 1.00e
+000 1.00e+000f 1
iter   objective   inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du
alpha_pr  ls
110 9.6287428e-001 0.00e+000 1.42e-002 -3.9 8.17e-003 -
9.73e-001 1.00e+000f 1
111 9.6354708e-001 0.00e+000 2.58e-002 -3.9 8.90e-003 - 1.00e
+000 1.00e+000f 1
112 9.6275569e-001 0.00e+000 1.57e-002 -3.9 3.30e-003 - 1.00e
+000 1.00e+000f 1
113 9.6411641e-001 0.00e+000 8.58e-003 -3.9 3.09e-003 - 1.00e
+000 1.00e+000f 1
114 9.6496170e-001 0.00e+000 9.81e-003 -3.9 2.33e-003 - 1.00e
+000 1.00e+000f 1
115 9.6544646e-001 0.00e+000 1.34e-002 -3.9 4.57e-003 - 1.00e
+000 1.00e+000f 1
116 9.6395176e-001 0.00e+000 9.50e-003 -3.9 3.21e-003 - 1.00e
+000 1.00e+000f 1
117 9.6240002e-001 0.00e+000 9.36e-003 -3.9 2.70e-003 - 1.00e
+000 1.00e+000f 1

```

```

118 9.6022179e-001 0.00e+000 1.17e-002 -3.9 5.04e-003 - 1.00e
+000 1.00e+000f 1
119 9.5834308e-001 0.00e+000 1.01e-002 -3.9 6.28e-003 - 1.00e
+000 1.00e+000f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
120 9.5692798e-001 0.00e+000 1.02e-002 -3.9 6.15e-003 - 1.00e
+000 1.00e+000f 1
121 9.5631828e-001 0.00e+000 9.43e-003 -3.9 2.57e-003 - 1.00e
+000 1.00e+000f 1
122 9.5589146e-001 0.00e+000 9.12e-003 -3.9 2.75e-003 - 1.00e
+000 1.00e+000f 1
123 9.5460755e-001 0.00e+000 1.25e-002 -3.9 4.73e-003 - 1.00e
+000 1.00e+000f 1
124 9.5356639e-001 0.00e+000 8.97e-003 -3.9 5.00e-003 - 1.00e
+000 1.00e+000f 1
125 9.5200129e-001 0.00e+000 7.66e-003 -3.9 4.14e-003 - 1.00e
+000 1.00e+000f 1
126 9.5026339e-001 0.00e+000 8.59e-003 -3.9 4.80e-003 - 1.00e
+000 1.00e+000f 1
127 9.4736202e-001 0.00e+000 1.45e-002 -3.9 8.28e-003 - 1.00e
+000 1.00e+000f 1
128 9.4458164e-001 0.00e+000 1.01e-002 -3.9 8.02e-003 - 1.00e
+000 1.00e+000f 1
129 9.3746848e-001 0.00e+000 1.60e-002 -4.5 1.96e-002 - 1.00e
+000 1.00e+000f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
130 9.3042564e-001 0.00e+000 1.71e-002 -5.5 3.35e-002 - 1.00e
+000 4.50e-001f 1
131 9.2861860e-001 0.00e+000 8.88e-002 -6.1 1.48e-002 - 1.00e
+000 1.31e-001f 1
132 9.2292080e-001 0.00e+000 3.43e-002 -11.0 1.47e-002 -
5.63e-001 3.58e-001f 1
133 9.1881489e-001 0.00e+000 2.61e-002 -5.2 1.03e-002 -
5.53e-001 2.69e-001f 1
134 9.1625790e-001 0.00e+000 8.07e-002 -5.8 1.20e-002 - 1.00e
+000 1.81e-001f 1
135 9.1240944e-001 0.00e+000 4.97e-002 -7.6 1.42e-002 -
4.05e-001 2.51e-001f 1
136 9.0989065e-001 0.00e+000 3.01e-002 -5.4 1.35e-002 -
4.33e-001 1.65e-001f 1
137 9.0875589e-001 0.00e+000 3.23e-002 -11.0 1.14e-002 -
4.26e-001 9.94e-002f 1
138 9.0651262e-001 0.00e+000 3.72e-002 -6.3 1.50e-002 - 1.00e
+000 1.65e-001f 1
139 9.0420033e-001 0.00e+000 2.63e-002 -5.7 1.48e-002 -
9.29e-001 1.98e-001f 1
iter objective inf_pr inf_du lg(mu) ||d|| lg(rg) alpha_du
alpha_pr ls
140 9.0300141e-001 0.00e+000 7.30e-002 -5.8 8.12e-003 -
9.36e-001 2.08e-001f 1
141 9.0058145e-001 0.00e+000 5.36e-002 -5.1 1.31e-002 -
9.96e-001 3.26e-001f 1

```

```

142 8.9954983e-001 0.00e+000 3.09e-002 -5.4 1.42e-002 -
6.63e-001 1.23e-001f 1
143 8.9807712e-001 0.00e+000 5.16e-002 -5.6 9.59e-003 -
7.45e-001 2.63e-001f 1
144 8.9639926e-001 0.00e+000 4.59e-002 -5.1 1.13e-002 -
8.20e-001 3.47e-001f 1
145 8.9499442e-001 0.00e+000 4.16e-002 -5.9 1.16e-002 -
5.68e-001 2.23e-001f 1
146 8.9383728e-001 0.00e+000 2.83e-002 -5.1 1.18e-002 -
6.35e-001 2.34e-001f 1
147 8.9237924e-001 0.00e+000 3.19e-002 -5.0 1.39e-002 -
8.53e-001 4.46e-001f 1
148 8.9159228e-001 0.00e+000 4.37e-002 -5.3 3.47e-003 -
9.06e-001 3.70e-001f 1
149 8.9053598e-001 0.00e+000 3.51e-002 -11.0 6.22e-003 -
4.21e-001 2.63e-001f 1
iter   objective   inf_pr   inf_du lg(mu)  ||d|| lg(rg) alpha_du
alpha_pr ls
150 8.8968815e-001 0.00e+000 4.40e-002 -6.9 8.64e-003 -
7.56e-001 1.77e-001f 1
151 8.8889024e-001 0.00e+000 5.08e-002 -7.1 1.00e-002 -
9.36e-001 1.58e-001f 1
152 8.8784494e-001 0.00e+000 5.08e-002 -5.6 8.97e-003 -
7.42e-001 2.58e-001f 1
153 8.8938157e-001 0.00e+000 1.26e-001 -4.8 1.86e-002 -
5.65e-001 9.14e-001f 1
154 8.8805800e-001 0.00e+000 1.04e-001 -5.0 2.05e-003 - 1.00e
+000 9.84e-001f 1
155 8.8761731e-001 0.00e+000 5.06e-002 -5.0 3.64e-003 -
7.33e-001 1.00e+000f 1
156 8.8708058e-001 0.00e+000 2.76e-002 -5.6 3.26e-003 -
8.48e-001 2.94e-001f 1
157 8.8602966e-001 0.00e+000 5.33e-002 -6.1 5.01e-003 -
8.43e-001 3.43e-001f 1
158 8.8544657e-001 0.00e+000 5.82e-002 -5.5 4.78e-003 -
9.74e-001 2.68e-001f 1
159 8.8471922e-001 0.00e+000 3.82e-002 -6.4 7.62e-003 -
6.93e-001 2.19e-001f 1
iter   objective   inf_pr   inf_du lg(mu)  ||d|| lg(rg) alpha_du
alpha_pr ls
160 8.8393186e-001 0.00e+000 3.39e-002 -6.6 1.34e-002 -
9.10e-001 1.61e-001f 1

```

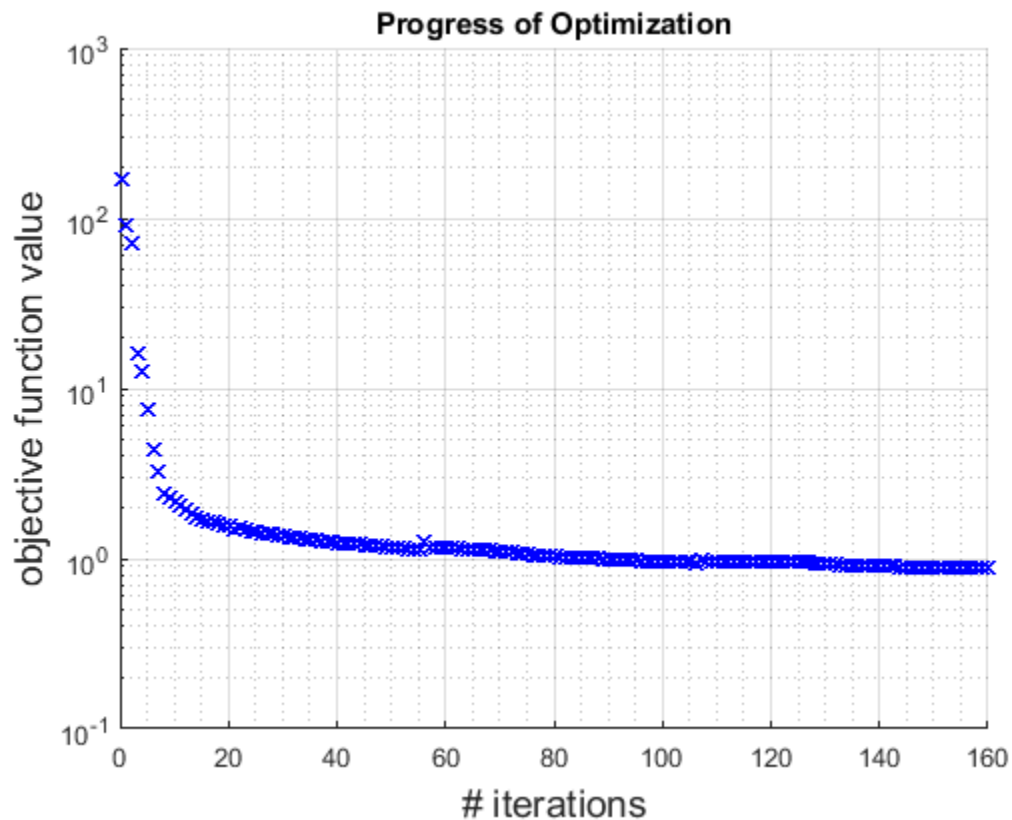
Number of Iterations.....: 160

	(scaled)	(unscaled)
Objective.....:	8.8393185574438515e-001	
	8.8393185574438515e-001	
Dual infeasibility.....:	3.3948503162460439e-002	
	3.3948503162460439e-002	
Constraint violation.....:	0.0000000000000000e+000	
	0.0000000000000000e+000	
Complementarity.....:	7.8777891974112479e-006	
	7.8777891974112479e-006	

Overall NLP error.....: 3.3948503162460439e-002
3.3948503162460439e-002

Number of objective function evaluations	=	161
Number of objective gradient evaluations	=	161
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)	=	4.618
Total CPU secs in NLP function evaluations	=	56.848

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

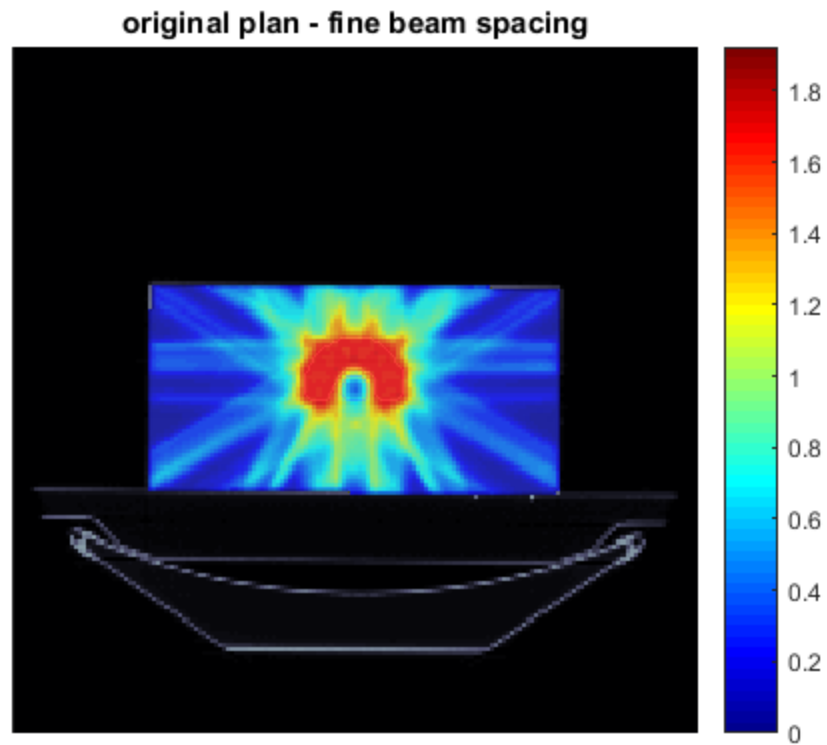


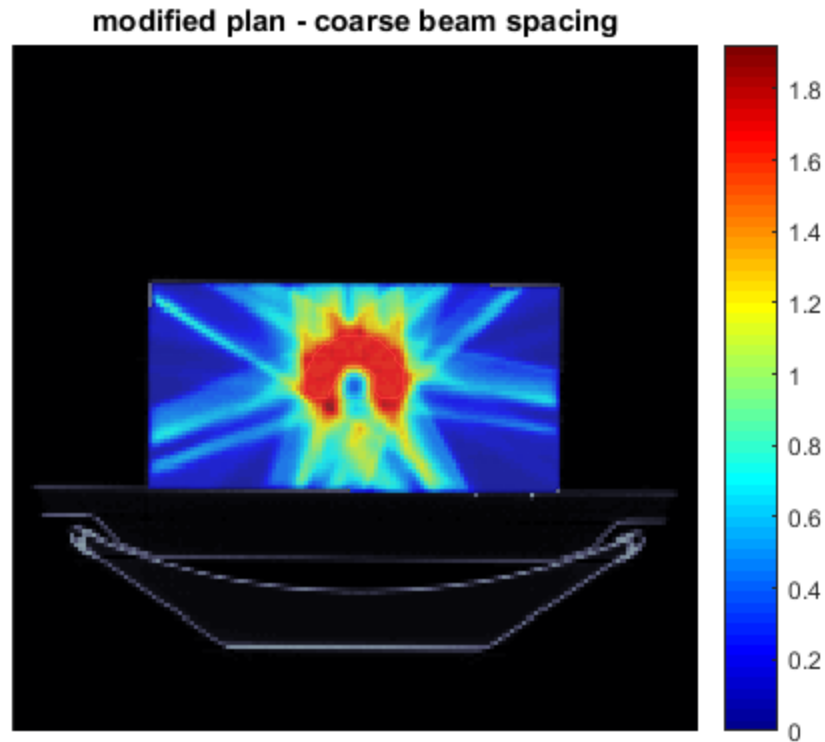
Visual Comparison of results

Let's compare the new recalculation against the optimization result. Check if you have added all SUB-DIRECTORIES to the MATLAB search path, otherwise it will not find the plotting function

```
plane      = 3;
doseWindow = [0 max([resultGUI.physicalDose(:);
    resultGUI_coarse.physicalDose(:)])];
```

```
figure,title('original plan - fine beam spacing')
[~,~,~,~,~] =
    matRad_plotSliceWrapper(gca,ct,cst,1,resultGUI.physicalDose,plane,slice,
    [],0.75,colorcube,[],doseWindow,[]);
figure,title('modified plan - coarse beam spacing')
[~,~,~,~,~] =
    matRad_plotSliceWrapper(gca,ct,cst,1,resultGUI_coarse.physicalDose,plane,slice,
    [],0.75,colorcube,[],doseWindow,[]);
```

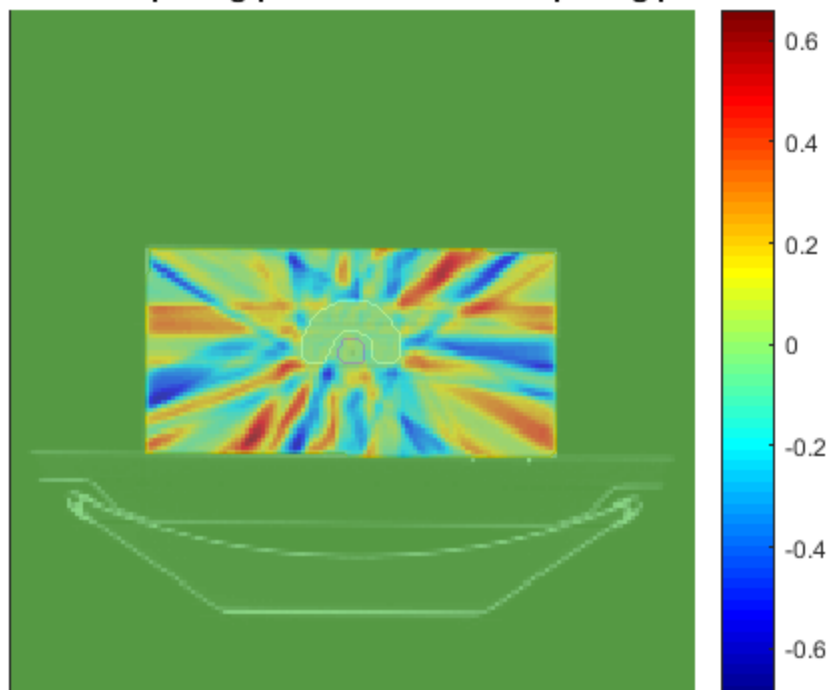




At this point we would like to see the absolute difference of the first optimization (finer beam spacing) and the second optimization (coarser beam spacing)

```
absDiffCube = resultGUI.physicalDose-resultGUI_coarse.physicalDose;
figure,title( 'fine beam spacing plan - coarse beam spacing plan')
[~,~,~,~,~] =
    matRad_plotSliceWrapper(gca,ct,cst,1,absDiffCube,plane,slice,[],
    [],colorcube);
```

fine beam spacing plan - coarse beam spacing plan



Obtain dose statistics

Two more columns will be added to the cst structure depicting the DVH and standard dose statistics such as D95,D98, mean dose, max dose etc.

```
cst          = matRad_indicatorWrapper(cst,pln,resultGUI);
cst_coarse   = matRad_indicatorWrapper(cst,pln,resultGUI_coarse);

0           BODY - Mean dose = 0.17 Gy +/- 0.33 Gy (Max dose
= 1.72 Gy, Min dose = 0.00 Gy)
              D2% = 1.39 Gy, D5% = 0.90 Gy, D98% =
0.00 Gy, D95% = 0.00 Gy,
              V0Gy = 100.00%, V0.345Gy = 17.58%, V0.69Gy
= 8.17%, V1.03Gy = 3.69%, V1.38Gy = 2.03%, V1.72Gy = 0.00%,

1           Core - Mean dose = 0.59 Gy +/- 0.18 Gy (Max dose
= 0.91 Gy, Min dose = 0.17 Gy)
              D2% = 0.91 Gy, D5% = 0.86 Gy, D98% =
0.21 Gy, D95% = 0.29 Gy,
              V0Gy = 100.00%, V0.345Gy = 91.95%, V0.69Gy
= 33.28%, V1.03Gy = 0.00%, V1.38Gy = 0.00%, V1.72Gy = 0.00%,

2           OuterTarget - Mean dose = 1.67 Gy +/- 0.02 Gy (Max dose
= 1.72 Gy, Min dose = 1.57 Gy)
              D2% = 1.70 Gy, D5% = 1.69 Gy, D98% =
1.60 Gy, D95% = 1.63 Gy,
```

$V0Gy = 100.00\%$, $V0.345Gy = 100.00\%$, $V0.69Gy = 100.00\%$, $V1.03Gy = 100.00\%$, $V1.38Gy = 100.00\%$, $V1.72Gy = 0.01\%$,
 $CI = 0.8581$, $HI = 3.44$ for reference dose
of 1.7 Gy

0 $BODY - \text{Mean dose} = 0.17 \text{ Gy} \pm 0.34 \text{ Gy}$ (Max dose = 1.92 Gy, Min dose = 0.00 Gy)
 $D2\% = 1.45 \text{ Gy}$, $D5\% = 0.94 \text{ Gy}$, $D98\% = 0.00 \text{ Gy}$, $D95\% = 0.00 \text{ Gy}$,
 $V0Gy = 100.00\%$, $V0.385Gy = 16.17\%$,
 $V0.769Gy = 6.85\%$, $V1.15Gy = 3.19\%$, $V1.54Gy = 1.73\%$, $V1.92Gy = 0.00\%$,

1 $Core - \text{Mean dose} = 0.60 \text{ Gy} \pm 0.17 \text{ Gy}$ (Max dose = 0.90 Gy, Min dose = 0.21 Gy)
 $D2\% = 0.89 \text{ Gy}$, $D5\% = 0.87 \text{ Gy}$, $D98\% = 0.24 \text{ Gy}$, $D95\% = 0.36 \text{ Gy}$,
 $V0Gy = 100.00\%$, $V0.385Gy = 90.55\%$,
 $V0.769Gy = 20.31\%$, $V1.15Gy = 0.00\%$, $V1.54Gy = 0.00\%$, $V1.92Gy = 0.00\%$,

2 $OuterTarget - \text{Mean dose} = 1.66 \text{ Gy} \pm 0.02 \text{ Gy}$ (Max dose = 1.72 Gy, Min dose = 1.54 Gy)
 $D2\% = 1.70 \text{ Gy}$, $D5\% = 1.70 \text{ Gy}$, $D98\% = 1.61 \text{ Gy}$, $D95\% = 1.63 \text{ Gy}$,
 $V0Gy = 100.00\%$, $V0.385Gy = 100.00\%$,
 $V0.769Gy = 100.00\%$, $V1.15Gy = 100.00\%$, $V1.54Gy = 100.00\%$, $V1.92Gy = 0.00\%$,
 $CI = 0.7568$, $HI = 3.76$ for reference dose
of 1.7 Gy

The treatment plan using more beams should in principle result in a better OAR sparing. Therefore lets have a look at the D95 of the OAR of both plans

```

ixOAR = 2;
cst{ixOAR,9}{1}.D95
cst_coarse{ixOAR,9}{1}.D95

```

ans =

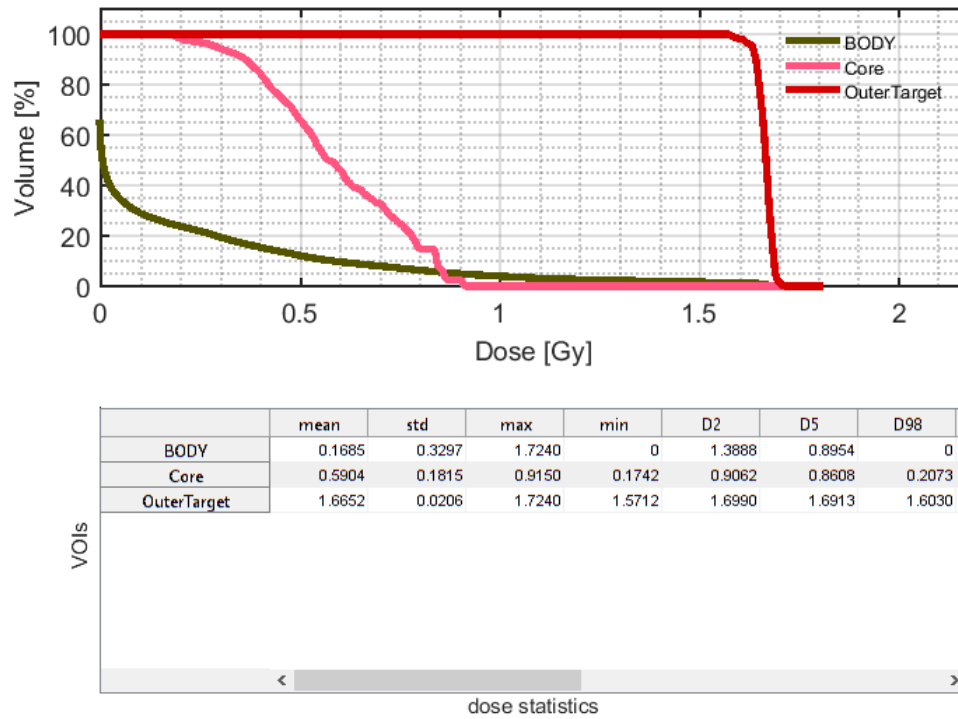
0.2926

ans =

0.3610

Indeed, the coarse beam plan yields a higher D95 in the OAR. Finally, lets plot the DVH

```
matRad_showDVH(cst,pln)
```



Published with MATLAB® R2017a