R2D2: A Dynamic Digital Phantom Documentation

Version 1.0

Chengyue Wu

September 09, 2020

Table of Contents

Welcome to the *R2D2* Dynamic Digital Phantom documentation!

This documentation provides information regarding how to download, visualize and use data involved in the *R2D2* Dynamic Digital Phantom, which is designed to mimic realistic perfusion and contrast agent delivery in kidney. It can be used as inputs and group-truth for validation and evaluation of imaging techniques, especially perfusion-based or contrast-enhanced modalities.

This phantom was developed by Dr. Thomas Yankeelov's group at Center for Computational Oncology (https://cco.oden.utexas.edu) at Oden Institute, The University of Texas at Austin. Readers are encouraged to refer to the associated manuscript for details on methods constructing the phantom. Further questions and discussion are welcome through the GitHub repository (), or *via* email (cw35926@utexas.edu).

Chapter 1 Introduction

1.1 Toolkit Download

All the data involved in this phantom toolkit are shared through Google Drive. Users are encouraged to download part of or the whole dataset according to their own needs.

Three are three major parts of the phantom shared in the toolkit, ‘Geometric Model’, ‘Steady-State Flow’ and ‘Contrast Agent Deliver’, each of which stored in a folder.

The folder ‘Geometric Model’ stores files storing information of geometry of the phantom, in particular, vasculature and kidney tissue. The folder ‘Steady-state Flow’ stores data of computed hemodynamic characteristics of the phantom, including interstitial pressure, interstitial flow velocity, blood pressure and vascular hydraulic conductivity. The folder ‘Contrast Agent Delivery’ stores data for the computed, spatiotemporally resolved distributions of tracer propagating through vasculature and interstitial tissue.

We are describing the data and how to visualize and use them in each folder in chapter below. Further questions and discussion are welcome *via* email, cw35926@utexas.edu.

1.2 Required Platforms

Data provided in this phantom include both regular-grid-based (i.e., voxelized) representations and unstructured-mesh-based representations of domain geometries, steady-state flow fields and time-dependent distribution of contrast agent.

The voxelized data are stored in MATLAB-files (i.e., *.mat) , where loading and visualization are tested to be successfully supported by MATLAB versions R2016b – R2020a. No specific MATLAB package is required for using the voxelized data, so earlier or later versions of MATLAB should also be compatible.

The mesh-based representations are stored as XML or HDF5 python files. Python-based computing platform, FEniCS Porject (https://fenicsproject.org/), is required for loading and further processing of these data.

All related demos and examples included in this documentation are built on FEniCS version 2017.2.0. Visualization of the mesh-based representations *via* ParaView is recommended. Examples of converting HDF5 data files to VTK image files as well as visualizing using ParaView version 5.6.0 are also included in this documentation.

Chapter 2 Geometric Model

The folder “Geometric Model” contains files defining the geometry and tissue properties of the phantom domains. These data are important as supports to loading and processing computed flow fields and contrast agent dynamics. Specifically, four files can be found in the folder:

1) **‘DigPhant_Geo_FinalMasks.mat’:**
   Segmentation mask defining tissue types in the phantom.
2) **‘DigPhant_Geo_VascularGraphs.mat’:**
   Graph defining centerlines and hierarchy structure of vasculature.
3) **‘DigPhant_TissueMesh_dimensionless.xml’:**
   3D mesh of interstitial (i.e., extravascular) tissue.
4) **‘DigPhant_VesselMesh_dimensionless.xml’:**
   3D mesh of vasculature.

We give detailed explanation for each file, as well as visualization demo in following sections.

2.1 ‘DigPhant_Geo_FinalMasks.mat’

The mask segmented tissue types in the phantom, in particular vascular and extravascular regions is stored as a map in MATLAB-file.

- ***Variables***

Variable Name	**Type**	**Notes**	
FOV_Size	double	Physical size of the entire 3D phantom, with unit = [mm]	
Masks	uint8	Labels for tissue types	
		= 0	void
		= 1	Interstitial tissue
		= 2	Arterial vessels

- ***Loading & Visualization via MATLAB script:***

```
load('DigPhant_Geo_FinalMasks.mat')

figure
imagesc(Masks(:,:,288))
axis equal tight off
colorbar
```

- ***Visualization Outputs:***

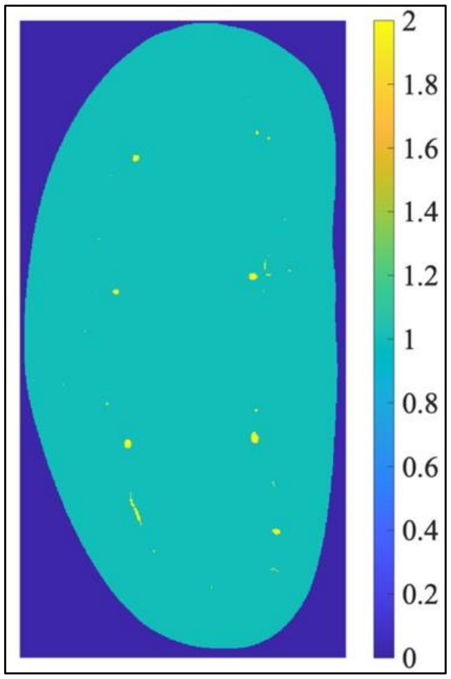


Figure 2.1
Mask defining tissue types in the phantom (visualized on the central slice). Vascular region = 2; Interstitial kidney tissue = 1; Void region = 0.

2.2 ‘DigPhant_Geo_VascularGraphs.mat’

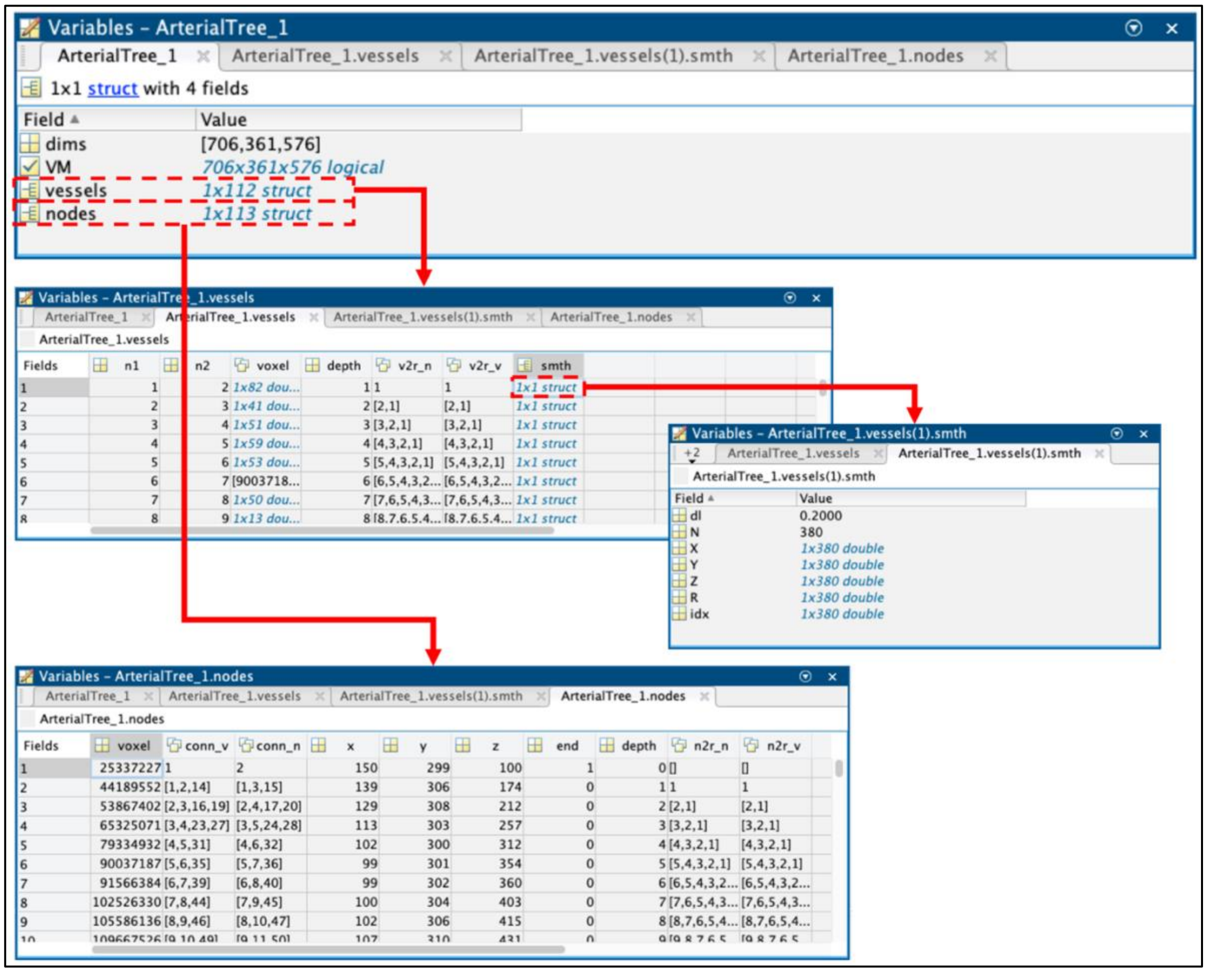
This MATLAB-file stores directed graphs that contain information of each constructed arterial tree and the whole vascular network, including the centerlines, local radius and branching structure.

- ***Variables:***

Variable Name	**Notes**		
ArterialTree_1	Data structure storing information of the 1st arterial vascular tree. Involved fields listed below --		
	.dims	matrix size of the phantom grid; = [706, 361, 576]	
	.VM	Binary map labeling voxels belonging to region of 1st arterial vascular tree.	
	.vessels	Data structure storing information of each vessel in the 1st vascular tree. Size = 1 × #(vessels), for each element storing information of one vessel. Involved fields listed below --	
		.n1	Node index of the starting point of this vessel (refers to the elements in ‘ArtertialTree_1.nodes’)
		.n2	Node index of the ending point of this vessel (refers to the elements in ‘ArtertialTree_1.nodes’)
		.voxel	Voxel indices of points along this vessel
		.depth	Hierarchy depth from the root of vascular tree to this vessel

		.v2r_n	Indices of nodes along the tracking path from this vessel up to the root of vascular tree	
		.v2r_v	Indices of vessels along the tracking path from this vessel up to the root of vascular tree	
		.smth	To obtain a refined geometry of vessel, the centerline of vessel is smoothed. The information of vessel after smoothing is stored in this field, which is again a data structure with following fields --	
			.dl	Spatial step-width of smoothing, with unit of [voxel] = 0.031 mm
			.N	Number of points along this vessel after smoothing
			.X	x-coordinates (i.e., 2nd dimension in the 3D image array) of points along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.Y	y-coordinates (i.e., 1st dimension in the 3D image array) of points along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.Z	z-coordinates (i.e., 3rd dimension in the 3D image array) of points along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.R	Local radius along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.idx	Global indices of points along this smoothed vessel in the whole vascular tree (terminal ends labeled as NaN)
	.nodes	Data structure storing information of each nodes (i.e., branch or terminal points) in the 1st vascular tree. Size = 1 × #(nodes), for each element represent one node. Involved fields listed below --		
		.voxel	Voxel index of this node	
		.conn_v	Indices of vessels connected to this node. (refers to the elements in 'ArtertialTree_1.vessels')	
		.conn_n	Indices of nodes that this node is connected to. (refers to the elements in 'ArtertialTree_1.nodes')	
		.comy	y-coordinate of this node, with unit of [voxel] = 0.031 mm	
		.comx	x-coordinate of this node, with unit of [voxel] = 0.031 mm	
		.comz	z-coordinate of this node, with unit of [voxel] = 0.031 mm	
		.end	Label indicating if the node is a branching or terminal point: branching = 0, terminal = 1	
		.depth	Hierarchy depth from the root of vascular tree to this node	
		.n2r_n	Indices of nodes on the tracking path from this node up to the root of vascular tree	
		.n2r_v	Indices of vessels on the tracking path from this node up to the root of vascular tree	
ArterialTree_2	Data structure saving the information of the 2nd arterial vascular tree. Fields the same as 'ArterialTree_1'			

ArterialTree_3	Data structure saving the information of the 3rd arterial vascular tree. Fields the same as ‘ArterialTree_1’
ArterialTree_4	Data structure saving the information of the 4th arterial vascular tree. Fields the same as ‘ArterialTree_1’
ArterialTree_5	Data structure saving the information of the 5th arterial vascular tree. Fields the same as ‘ArterialTree_1’
ArterialTree_6	Data structure saving the information of the 6th arterial vascular tree. Fields the same as ‘ArterialTree_1’
VesselNetwork	Data structure saving the information of the entire vascular network. Fields the same as ‘ArterialTree_1’


Variables – ArterialTree_1
ArterialTree_1
ArterialTree_1.vessels
ArterialTree_1.vessels(1).smth
ArterialTree_1.nodes

1x1 struct with 4 fields

Field	Value
dims	[706,361,576]
VM	706x361x576 logical
vessels	1x112 struct
nodes	1x113 struct

Variables – ArterialTree_1.vessels

ArterialTree_1.vessels

Fields	n1	n2	voxel	depth	v2r_n	v2r_v	smth
1	1	2	1x82 dou...	1	1	1	1x1 struct
2	2	3	1x41 dou...	2	[2,1]	[2,1]	1x1 struct
3	3	4	1x51 dou...	3	[3,2,1]	[3,2,1]	1x1 struct
4	4	5	1x59 dou...	4	[4,3,2,1]	[4,3,2,1]	1x1 struct
5	5	6	1x53 dou...	5	[5,4,3,2,1]	[5,4,3,2,1]	1x1 struct
6	6	7	[9003718...	6	[6,5,4,3,2...	[6,5,4,3,2...	1x1 struct
7	7	8	1x50 dou...	7	[7,6,5,4,3...	[7,6,5,4,3...	1x1 struct
8	8	9	1x13 dou...	8	[8,7,6,5,4...	[8,7,6,5,4...	1x1 struct

ArterialTree_1.vessels(1).smth

Field	Value
dl	0.2000
N	380
X	1x380 double
Y	1x380 double
Z	1x380 double
R	1x380 double
idx	1x380 double

ArterialTree_1.nodes

Fields	voxel	conn_v	conn_n	x	y	z	end	depth	n2r_n	n2r_v
1	25337227	1	2	150	299	100	1	0	[]	[]
2	44189552	[1,2,14]	[1,3,15]	139	306	174	0	1	1	1
3	53867402	[2,3,16,19]	[2,4,17,20]	129	308	212	0	2	[2,1]	[2,1]
4	65325071	[3,4,23,27]	[3,5,24,28]	113	303	257	0	3	[3,2,1]	[3,2,1]
5	79334932	[4,5,31]	[4,6,32]	102	300	312	0	4	[4,3,2,1]	[4,3,2,1]
6	90037187	[5,6,35]	[5,7,36]	99	301	354	0	5	[5,4,3,2,1]	[5,4,3,2,1]
7	91566384	[6,7,39]	[6,8,40]	99	302	360	0	6	[6,5,4,3,2...	[6,5,4,3,2...
8	102526330	[7,8,44]	[7,9,45]	100	304	403	0	7	[7,6,5,4,3...	[7,6,5,4,3...
9	105586136	[8,9,46]	[8,10,47]	102	306	415	0	8	[8,7,6,5,4...	[8,7,6,5,4...
10	109667526	[9,10,49]	[9,11,50]	107	310	431	0	9	[9,8,7,6,5	[9,8,7,6,5

Figure 2.2

Screenshots showing structure of variables available in the file storing vascular geometry.

- ***Loading & Visualization via MATLAB scripts:***

```
load('DigPhant_Geo_VascularGraphs.mat');

close all
for ID = 1:6
eval(['vessels = ArterialTree_',num2str(ID),'.vessels;'])

    figure
    for l = 1:length(vessels)
        X = vessels(l).smth.X;
        Y = vessels(l).smth.Y;
        Z = vessels(l).smth.Z;

        plot3(X, Y, Z,'r', 'Linewidth',2)
        hold on
    end
    axis equal tight; grid on
    xlabel('x-axis'); ylabel('y-axis'); zlabel('z-axis')
    ax = gca(gcf); ax.FontSize = 16;
end
```

- ***Visualization Outputs:***

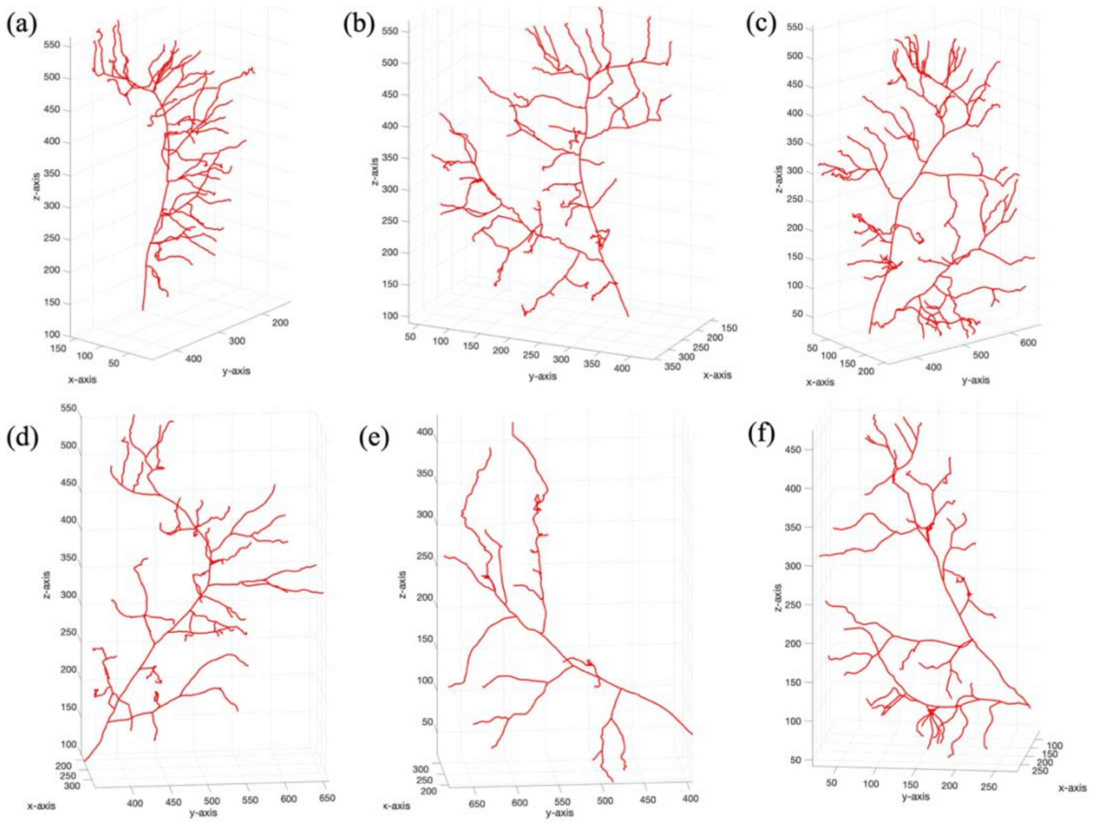


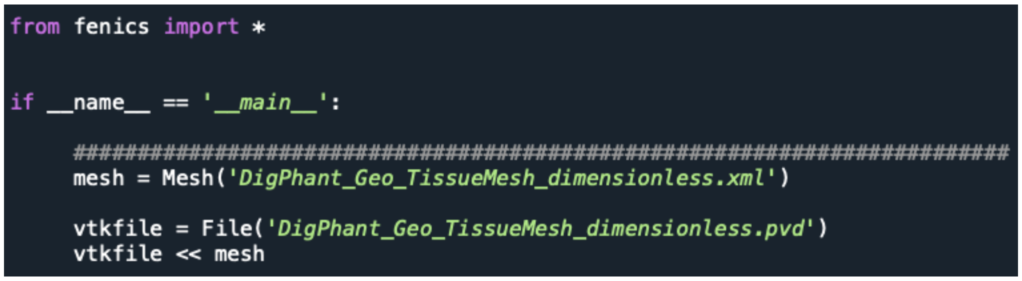
Figure 2.3

Visualizing the centerlines of vascular tree. Panels (a – f) represent the geometry of individual arterial tree, respectively.

2.3 ‘DigPhant_Geo_TissueMesh_dimensionless.xml’

- ***Loading via python script:***

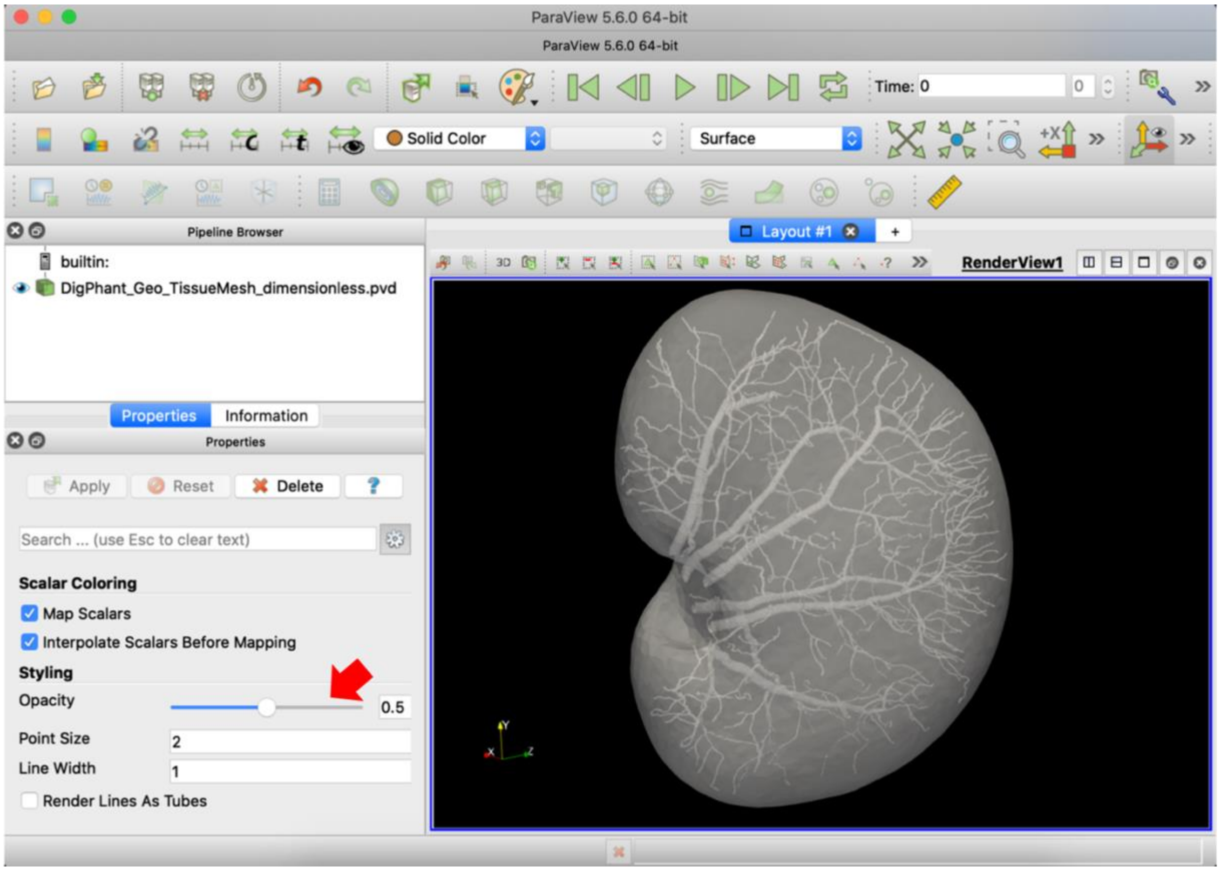
Using the class ‘Mesh’ defined in FEniCS, we can load the .xml file of 3D tissue mesh and then save it to VTK image file for visualization.



```
from fenics import *

if __name__ == '__main__':

    ##########################################################################
    mesh = Mesh('DigPhant_Geo_TissueMesh_dimensionless.xml')

    vtkfile = File('DigPhant_Geo_TissueMesh_dimensionless.pvd')
    vtkfile << mesh
```

- ***Visualization via ParaView:***

The boundary of this mesh consists of a contour of the kidney and the exterior surface of vasculature. Specifically, opening the VTK file of the 3D mesh of the interstitial tissue in ParaView, we can set the property ‘Surface Opacity’ of the view to be 0.5 (as indicated by red arrow in **Fig 2.4**), so that to see the vascular trajectories cutting through the mesh volume.



ParaView 5.6.0 64-bit

Pipeline Browser
builtin:
DigPhant_Geo_TissueMesh_dimensionless.pvd

Properties
Search ... (use Esc to clear text)

Scalar Coloring
☑ Map Scalars
☑ Interpolate Scalars Before Mapping

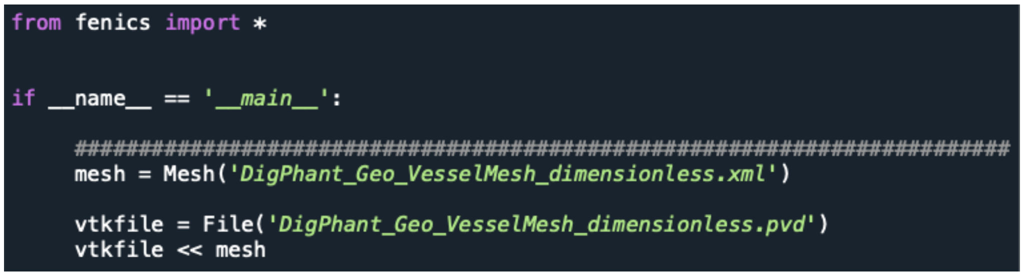
Styling
Opacity 0.5
Point Size 2
Line Width 1
☐ Render Lines As Tubes

Figure 2.4 Screenshot of visualizing 3D tissue mesh in ParaView.

2.4 ‘DigPhant_Geo_VesselMesh_dimensionless.xml’

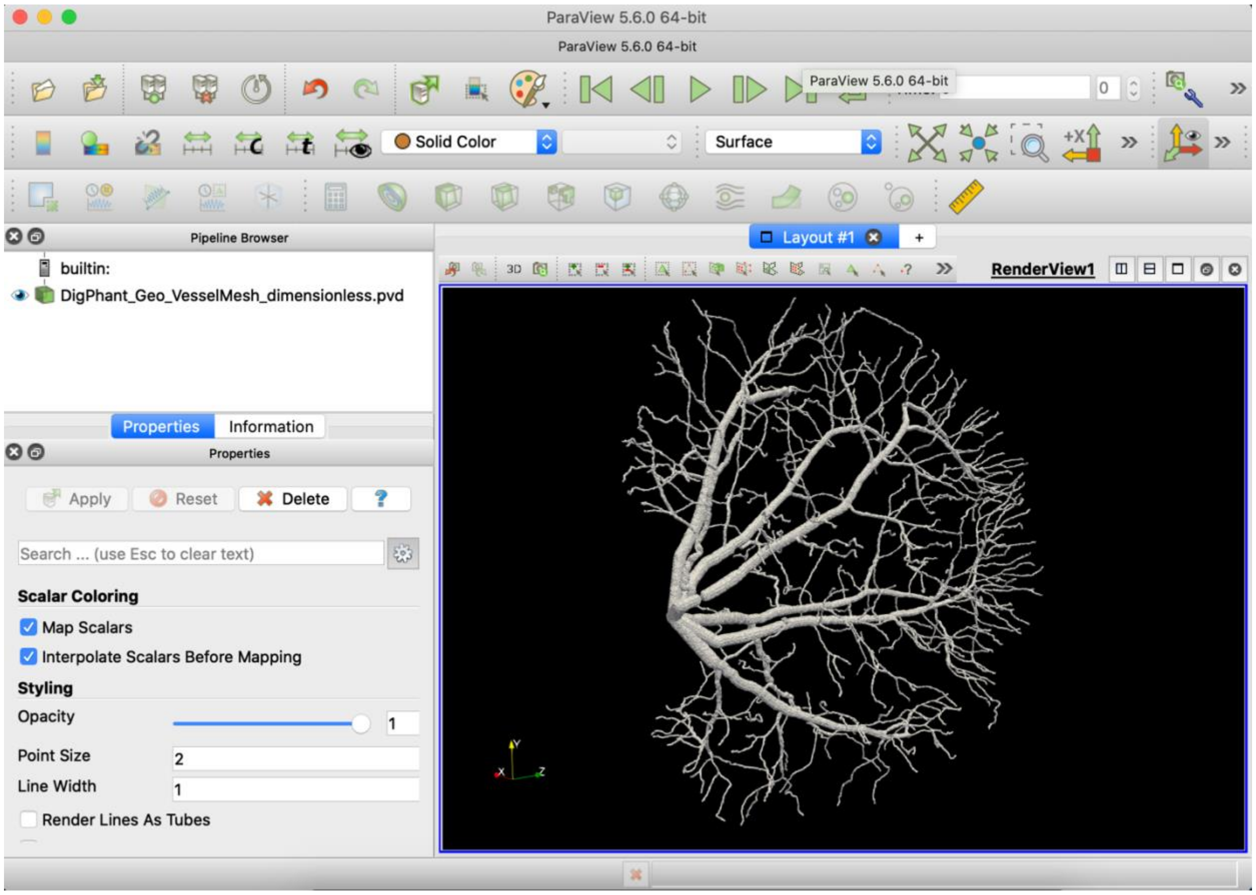
- ***Loading via python script:***

Similarly, using the class ‘Mesh’ defined in FEniCS, we can also load the .xml file of 3D vessel mesh and then save it to VTK image file for visualization.



```python
from fenics import *

if __name__ == '__main__':

    #########################################################################
    mesh = Mesh('DigPhant_Geo_VesselMesh_dimensionless.xml')

    vtkfile = File('DigPhant_Geo_VesselMesh_dimensionless.pvd')
    vtkfile << mesh
```

- ***Visualization via ParaView:***



ParaView 5.6.0 64-bit

Pipeline Browser

builtin:

DigPhant_Geo_VesselMesh_dimensionless.pvd

Properties

Scalar Coloring

- [x] Map Scalars
- [x] Interpolate Scalars Before Mapping

Styling

Opacity: 1

Point Size: 2

Line Width: 1

- [ ] Render Lines As Tubes

Figure 2.5 Screen shot of visualizing 3D vessel mesh in ParaView.

Chapter 3 Steady-state Flow

The folder “Steady-state Flow” contains files defining the computed solution of blood and interstitial flow fields. Specifically, five files can be found in the folder:

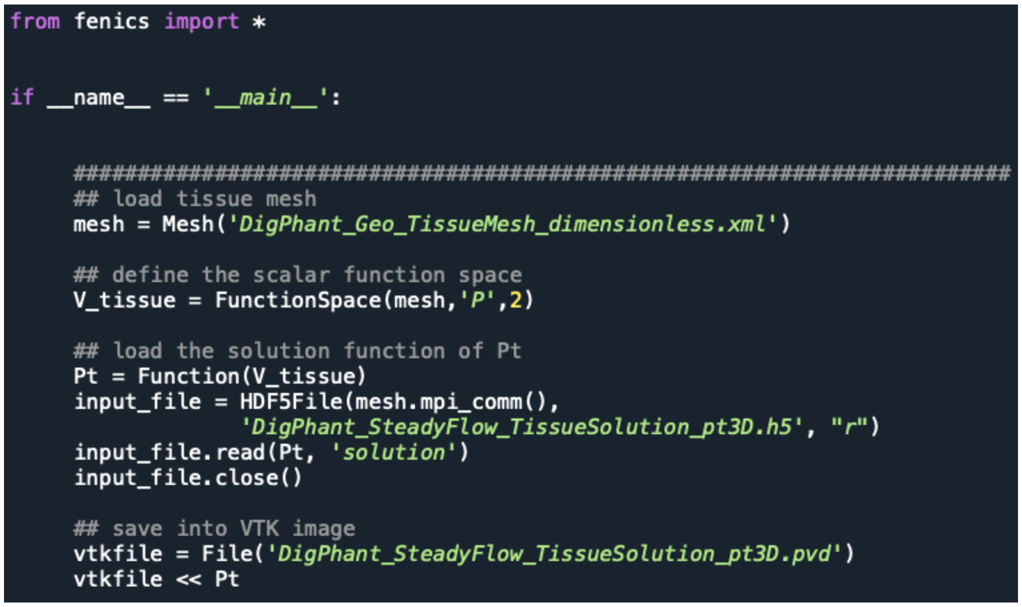
1) **‘DigPhant_SteadyFlow_TissueSolution_pt3D.h5’:**
   3D mesh-based solution of computed interstitial pressure field.
2) **‘DigPhant_SteadyFlow_TissueSolution_pt3D.mat’:**
   Voxel-wised distribution of computed interstitial pressure field.
3) **‘DigPhant_SteadyFlow_TissueSolution_ut3D.h5’:**
   3D mesh-based solution of computed interstitial flow velocity.
4) **‘DigPhant_SteadyFlow_TissueSolution_ut3D.mat’:**
   Voxel-wised distribution of computed interstitial flow velocity.
5) **‘DigPhant_SteadyFlow_VascularSolution.mat’**
   Graph-based (i.e., pseudo-1D) solution of computed blood flow.

We give detailed explanation for each file, as well as usage demo in following sections.

3.1 ‘DigPhant_SteadyFlow_TissueSolution_pt3D.h5’

The interstitial pressure field is numerically solved on a finite element scalar function space with the second-order polynomial basis. The solution is stored as a mesh-based, FEniCS-defined Function on the given function space. Note that to load the mesh-based representation of flow solution need the mesh files as preparation.

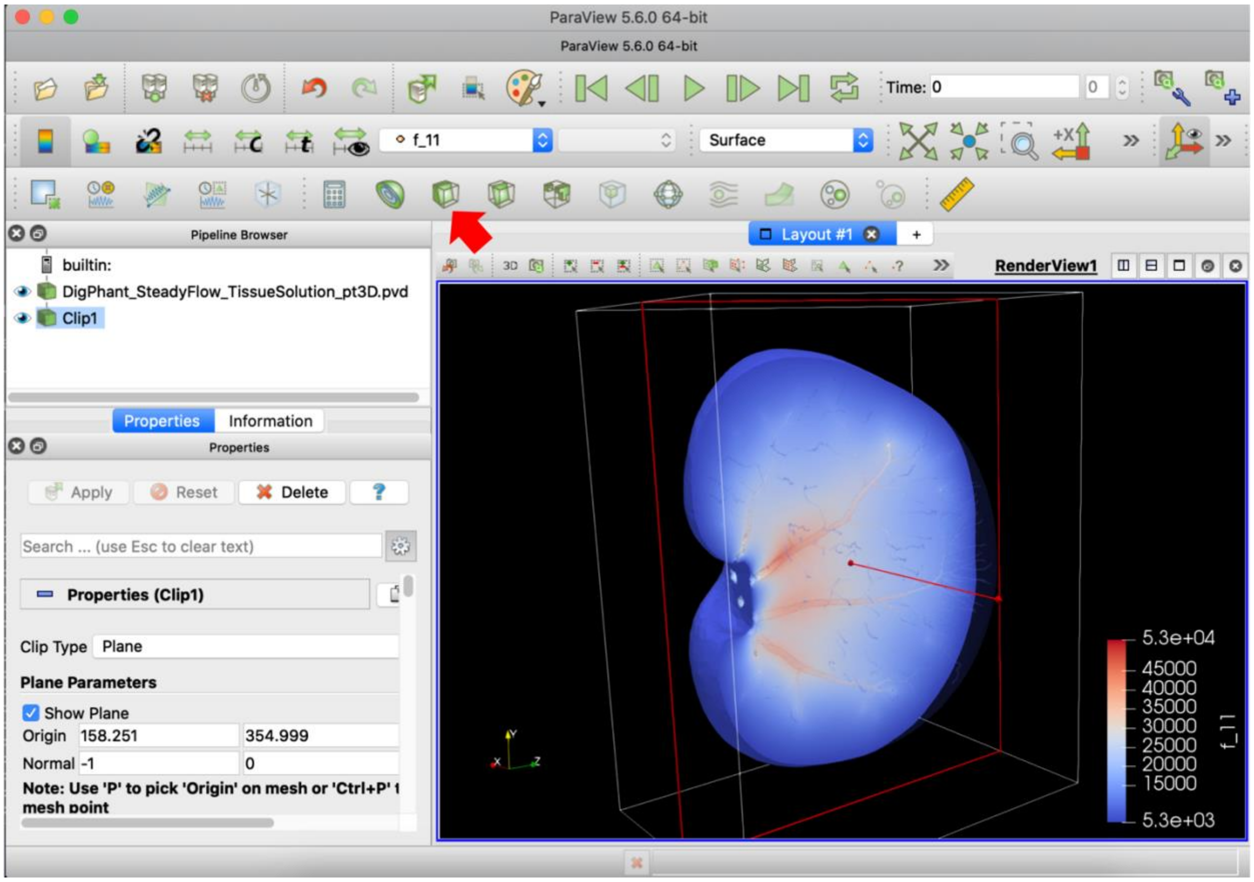
- ***Loading via python scripts:***



```
from fenics import *


if __name__ == '__main__':


    ########################################################################
    ## load tissue mesh
    mesh = Mesh('DigPhant_Geo_TissueMesh_dimensionless.xml')

    ## define the scalar function space
    V_tissue = FunctionSpace(mesh,'P',2)

    ## load the solution function of Pt
    Pt = Function(V_tissue)
    input_file = HDF5File(mesh.mpi_comm(),
                 'DigPhant_SteadyFlow_TissueSolution_pt3D.h5', "r")
    input_file.read(Pt, 'solution')
    input_file.close()

    ## save into VTK image
    vtkfile = File('DigPhant_SteadyFlow_TissueSolution_pt3D.pvd')
    vtkfile << Pt
```

- ***Visualization via ParaView:***

As shown in Fig 3.1, we open the VTK file of the mesh-based interstitial pressure solution in ParaView. To visualize the distribution of pressure within the kidney tissue, we set the property ‘Surface Opacity’ of the original 3D volume to be 0.5 and apply the operator ‘Clip’ (as indicated by the red arrow in Fig 3.1) on the 3D volume to generate a new chopped view.



builtin:
DigPhant_SteadyFlow_TissueSolution_pt3D.pvd
Clip1

Search ... (use Esc to clear text)

Properties (Clip1)

Clip Type: Plane

Plane Parameters

Show Plane ✓

Origin: 158.251, 354.999

Normal: -1, 0

Note: Use 'P' to pick 'Origin' on mesh or 'Ctrl+P' t mesh point

Figure 3.1

Screen shot of visualizing 3D mesh-based solution of interstitial pressure in ParaView.

3.2 'DigPhant_SteadyFlow_TissueSolution_pt3D.mat'

The numerical solution of interstitial pressure is converted into a voxel-wised distribution map with the size of 706 × 361 × 576 and saved in a MATLAB-file.

- ***Variables:***

Variable Name	Type	Notes
Pt	double	Magnitude of interstitial pressure, unit = [g cm^{-1} s^{-2}]

- ***Loading & Visualization via MATLAB script:***

```
load('DigPhant_SteadyFlow_TissueSolution_pt3D.mat')

figure
imagesc(Pt(:,:,288))
axis equal tight off
colormap(jet)
colorbar
```

- ***Visualization Outputs:***

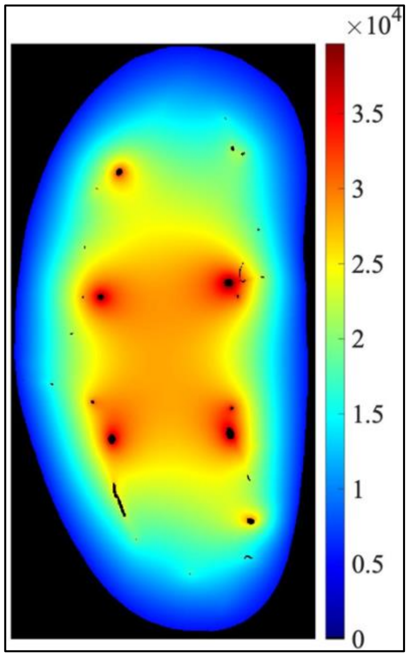
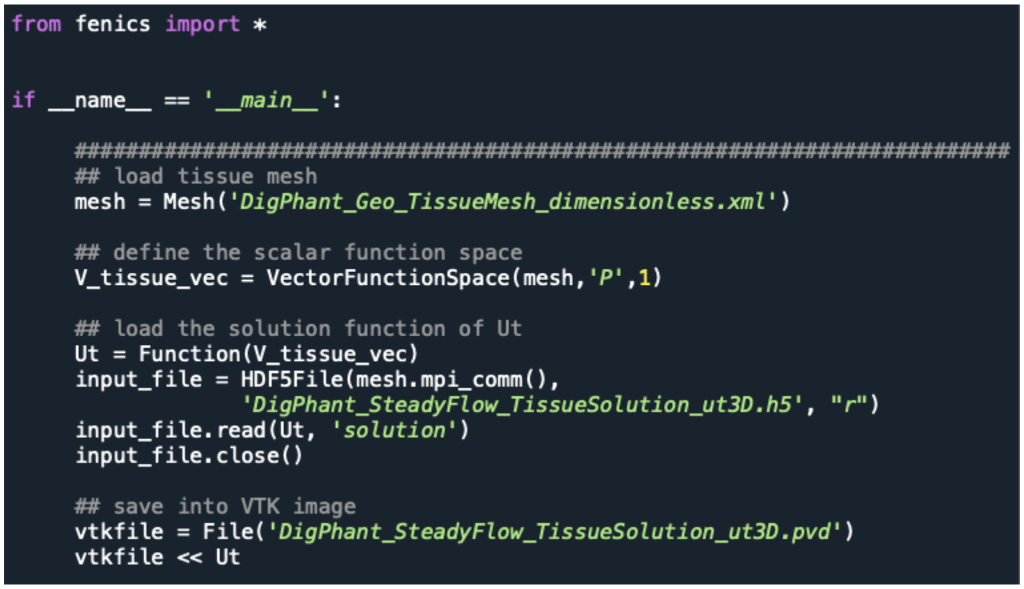


Figure 3.2
Interstitial pressure (Pt) map (visualized on the central slice). The heat color indicates the magnitude of pressure, with the unit of [g cm^{-1} s^{-2}].

3.3 ‘DigPhant_SteadyFlow_TissueSolution_ut3D.h5’

The interstitial flow velocity field is numerically solved on a finite element vector function space, with the first-order polynomial basis. The solution is stored as a mesh-based, FEniCS-defined Function on the given function space. Note that to load the mesh-based representation of flow solution need the mesh files as preparation.

- ***Loading via python script:***



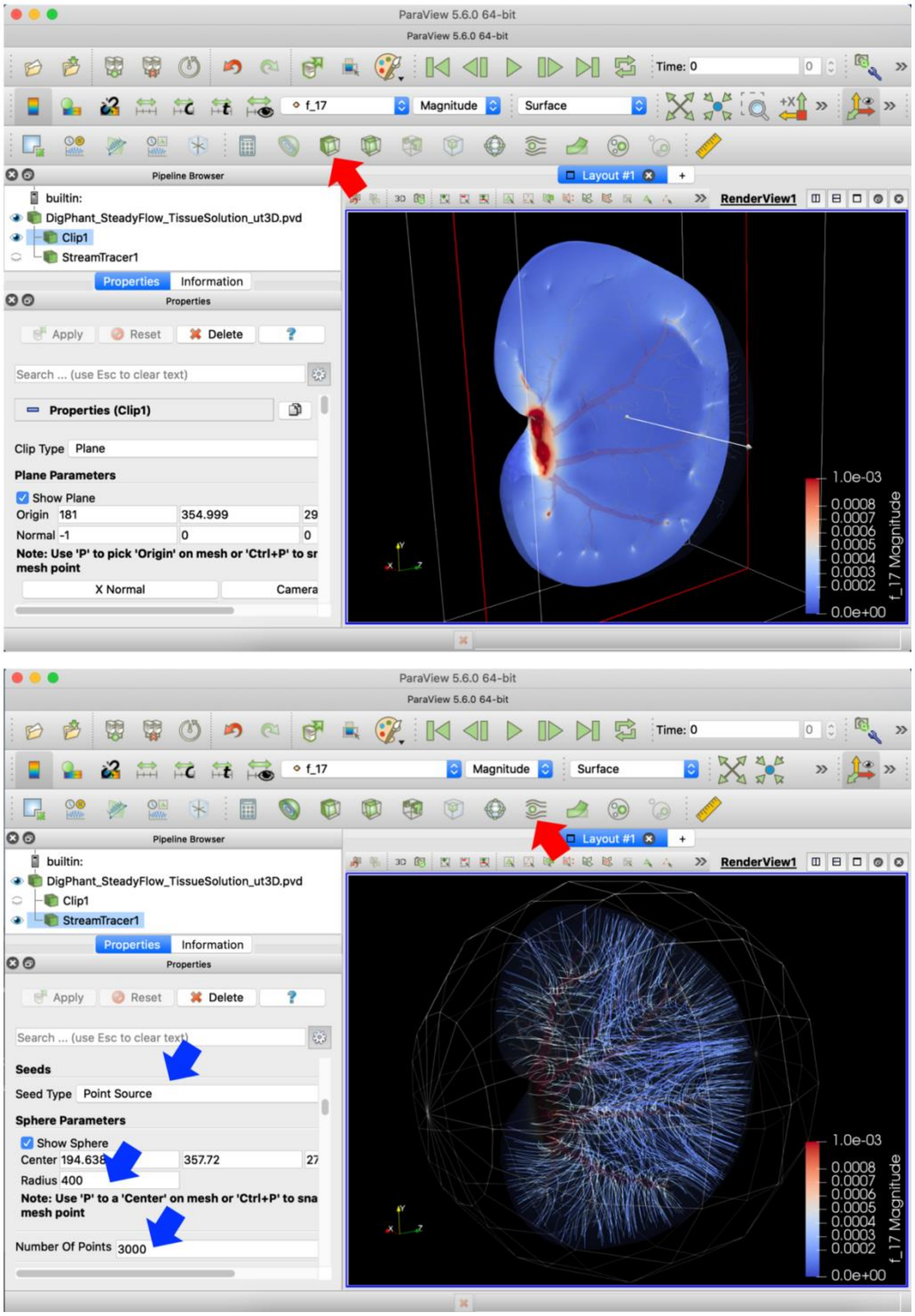
```python
from fenics import *


if __name__ == '__main__':

    ##########################################################################
    ## load tissue mesh
    mesh = Mesh('DigPhant_Geo_TissueMesh_dimensionless.xml')

    ## define the scalar function space
    V_tissue_vec = VectorFunctionSpace(mesh,'P',1)

    ## load the solution function of Ut
    Ut = Function(V_tissue_vec)
    input_file = HDF5File(mesh.mpi_comm(),
                 'DigPhant_SteadyFlow_TissueSolution_ut3D.h5', "r")
    input_file.read(Ut, 'solution')
    input_file.close()

    ## save into VTK image
    vtkfile = File('DigPhant_SteadyFlow_TissueSolution_ut3D.pvd')
    vtkfile << Ut
```

- ***Visualization via ParaView:***

As shown in Fig 3.3, we open the VTK file of mesh-based interstitial flow velocity solution in ParaView. In panel (a), to visualize the magnitude of flow velocity within the tissue, we set the property ‘Surface Opacity’ of the original 3D volume to be 0.3 and apply the operator ‘Clip’ (as indicated by the red arrow in Fig 3.3 (a)) on the 3D volume, so that to generate the a new chopped view. In panel (b), to visualize the direction and trajectory of flow, we apply the operator ‘Streamline’ (indicated by the red arrow in Fig 3.3 (b)) on the original 3D volume, setting the seed property as ‘Point Source’ with ‘*Radius*’ = 400 and ‘Number of Sampling Point’ = 3000 (as indicated by the blue arrows in Fig 3.3 (b)).

(a)

(b)



(a)

```
builtin:
DigPhant_SteadyFlow_TissueSolution_ut3D.pvd
  Clip1
  StreamTracer1
Properties   Information
Apply   Reset   Delete   ?
Search ... (use Esc to clear text)
Properties (Clip1)
Clip Type   Plane
Plane Parameters
[✓] Show Plane
Origin  181     354.999   29
Normal  -1      0         0
Note: Use 'P' to pick 'Origin' on mesh or 'Ctrl+P' to sr
mesh point
X Normal      Camera
```

(b)

```
builtin:
DigPhant_SteadyFlow_TissueSolution_ut3D.pvd
  Clip1
  StreamTracer1
Properties   Information
Apply   Reset   Delete   ?
Search ... (use Esc to clear text)
Seeds
Seed Type   Point Source
Sphere Parameters
[✓] Show Sphere
Center  194.638   357.72   27
Radius  400
Note: Use 'P' to a 'Center' on mesh or 'Ctrl+P' to sna
mesh point
Number Of Points  3000
```

Figure 3.3

Screenshots of visualizing 3D mesh-based solution of interstitial flow velocity in ParaView. Panel (a) emphasizes the magnitude with a ‘Clip’ view and panel (b) shows the streamlines of flow.

3.4 ‘DigPhant_SteadyFlow_TissueSolution_ut3D.mat’

The numerical solution of interstitial flow velocity is then converted into three voxel-wised distribution maps for velocity on x-, y- and z-directions, respectively and saved in a MATLAB-file, where each map has a size of 706 × 361 × 576.

- ***Variables:***

Variable Name	Type	Notes
Ut_x	double	Magnitude of interstitial flow velocity on the x-direction (i.e., 2nd dimension of the image array), unit = [cm / s]
Ut_y	double	Magnitude of interstitial flow velocity on the y-direction (i.e., 1st dimension of the image array), unit = [cm / s]
Ut_z	double	Magnitude of interstitial flow velocity on the z-direction (i.e., 3rd dimension of the image array), unit = [cm / s]

- ***Loading & Visualization via MATLAB script:***

```
load('DigPhant_Geo_FinalMasks.mat')
load('DigPhant_SteadyFlow_TissueSolution_ut3D.mat')

Mask_tissue = double(Masks == 1);
Mask_tissue(Mask_tissue == 0) = NaN;

Ut_x = Ut_x .* Mask_tissue;
Ut_y = Ut_y .* Mask_tissue;
Ut_z = Ut_z .* Mask_tissue;

c = jet;
c(1,:) = [0,0,0];

figure
subplot(1,3,1)
imagesc(Ut_x(:,:,288))
axis equal tight off
colormap(c)
colorbar
caxis([-1e-3, 1e-3])

subplot(1,3,2)
imagesc(Ut_y(:,:,288))
axis equal tight off
colormap(c)
colorbar
caxis([-1e-3, 1e-3])

subplot(1,3,3)
imagesc(Ut_z(:,:,288))
axis equal tight off
colormap(c)
colorbar
caxis([-1e-3, 1e-3])
```

- ***Visualization Outputs:***

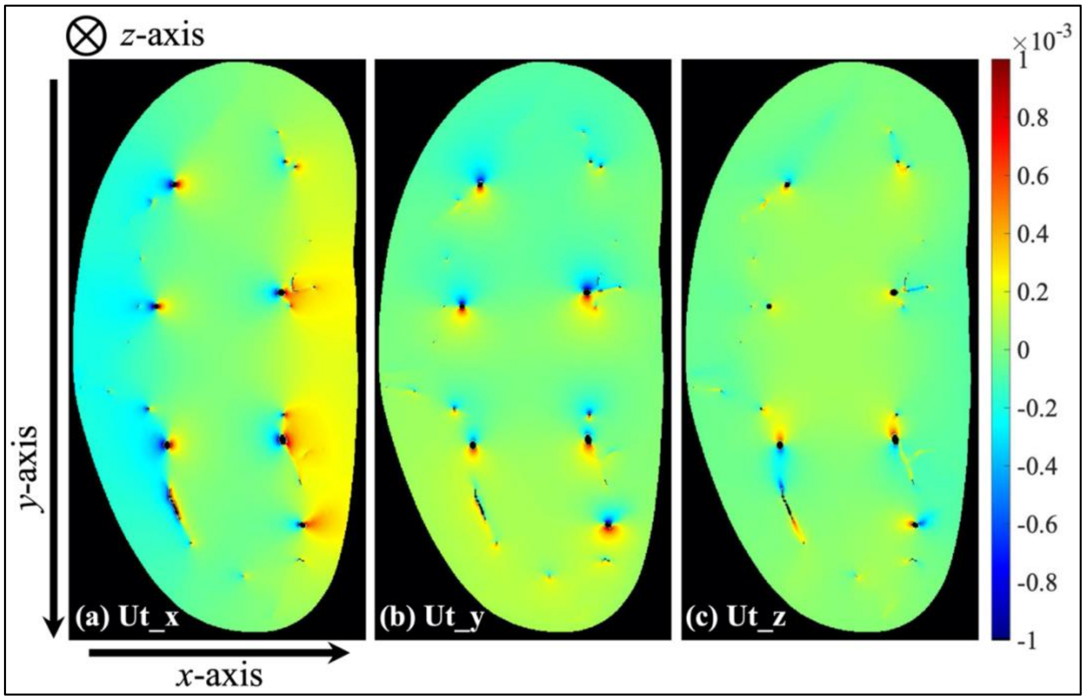


Figure 3.4

Interstitial flow velocity (Ut) map (visualized on the central slice). Panels (a – c) show computed velocity on x-, y- and z-directions, respectively, with the unit of [cm / s]. The heat color represents both the direction and magnitude of velocity, where the warm color indicates flow on the positive direction of an axis while the cold color indicates negative flow direction compared to the axis.

3.5 'DigPhant_SteadyFlow_VascularSolution.mat'

The blood pressure is solved as pseudo-1D along the centerlines of vascular network. This MATLAB file stores information and solutions along each vessel.

- ***Variables:***

Variable Name	Type	Notes
VesselNetwork_dl	double	Solution step-width of each vessel, with unit of [voxel] = 0.031 mm. Indices of element referring to the indices of vessels in the entire vascular network (i.e., variable "VesselNetwork" in file "**DigPhant_Geo_VascularGraphs.mat**"
VesselNetwork_Lp	cell	Vascular hydraulic conductivity along each vessel, with same indices as "VesselNetwork_dl" and unit = [g^{-1} cm^2 s]
VesselNetwork_N	int64	Number of solution points for each vessel, with same indices as "VesselNetwork_dl"
VesselNetwork_ptev	cell	Solution of exterior surface pressure along each vessel, with same indices as "VesselNetwork_dl" and unit = [g cm^{-1} s^{-2}]
VesselNetwork_pv	cell	Solution of blood pressure along each vessel, with same indices as "VesselNetwork_dl" and unit = [g cm^{-1} s^{-2}]

- ***Loading & Visualization via MATLAB script:***

As an example, we provide the script to plot the blood pressure solution at a bifurcation in the vascular network.

```
%%%% load in
load('DigPhant_Geo_VascularGraphs.mat');
load('DigPhant_SteadyFlow_VascularSolution.mat')

vessels = VesselNetwork.vessels;

close all
figure
%%%% visualize the vessel centerlines
subplot(1,2,1)
for l = [1,2 14]
    X = vessels(l).smth.X;
    Y = vessels(l).smth.Y;
    Z = vessels(l).smth.Z;

    plot3(X, Y, Z,'Linewidth',2)

    hold on
end
legend({'vessel no.1','vessel no.2','vessel no.14'})
axis equal tight; grid on
xlabel('x-axis'); ylabel('y-axis'); zlabel('z-axis')
ax = gca(gcf); ax.FontSize = 16;

%%%% plot vascular solutions along centerlines
subplot(1,2,2)
for l = [1,2,14]
    Pv_Value = VesselNetwork_pv{l};
    plot(Pv_Value,'Linewidth',2)
    hold on
end
legend({'vessel no.1','vessel no.2','vessel no.14'})
xlabel('point along vessel')
ylabel('blood pressure (p_v) [g cm^{-1} s^{-2}]')
ax = gca(gcf); ax.FontSize = 16;
```

- ***Visualization Outputs:***

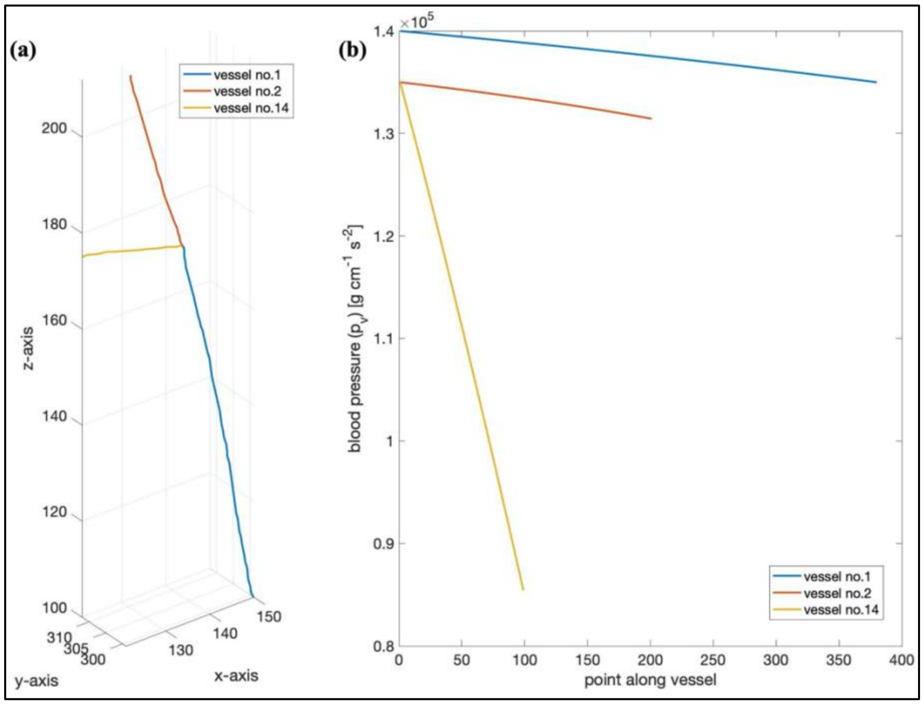


Figure 3.5

Panel (a) shows the centerline geometry of vessels in a bifurcation. Panel (b) plots the solved blood pressure along each of the vessels.

Chapter 4 Contrast Agent Delivery

The folder ‘Contrast Agent Deliver’ contains data of computed, time-dependent distributions of contrast agent in vascular, tissue, and entire domain. Specifically, there are one isolated file and two subfolders containing multiple files in the folder:

1) **‘DigPhant_CADelivery_BolusPropagation’:**
   Graphic data storing information of contrast agent propagating through the vessels.
2) **Subfolder: ‘H5_Tissue’:**
   Each file in this subfolder contains the spatial-resolved distribution of contrast agent in the interstitial tissue domain at one specific time point, stored as a mesh-based representation.
3) **Subfolder: ‘Mat_wholeDomain’:**
   Each file in this subfolder contains the spatial-resolved distribution of contrast agent in the whole kidney domain at one specific time point, stored as a voxelized map.

We give detailed explanation for each file, as well as visualization demo in following sections.

4.1 ‘DigPhant_CADelivery_BolusPropagation.mat’

The propagation of contrast agent is also solved along the pseudo-1D vascular centerlines. The format of information saved in this MATLAB-file is similar to that of the vascular geometry file, ‘DigPhant_Geo_VascularGraphs.mat’.

- ***Variables:***

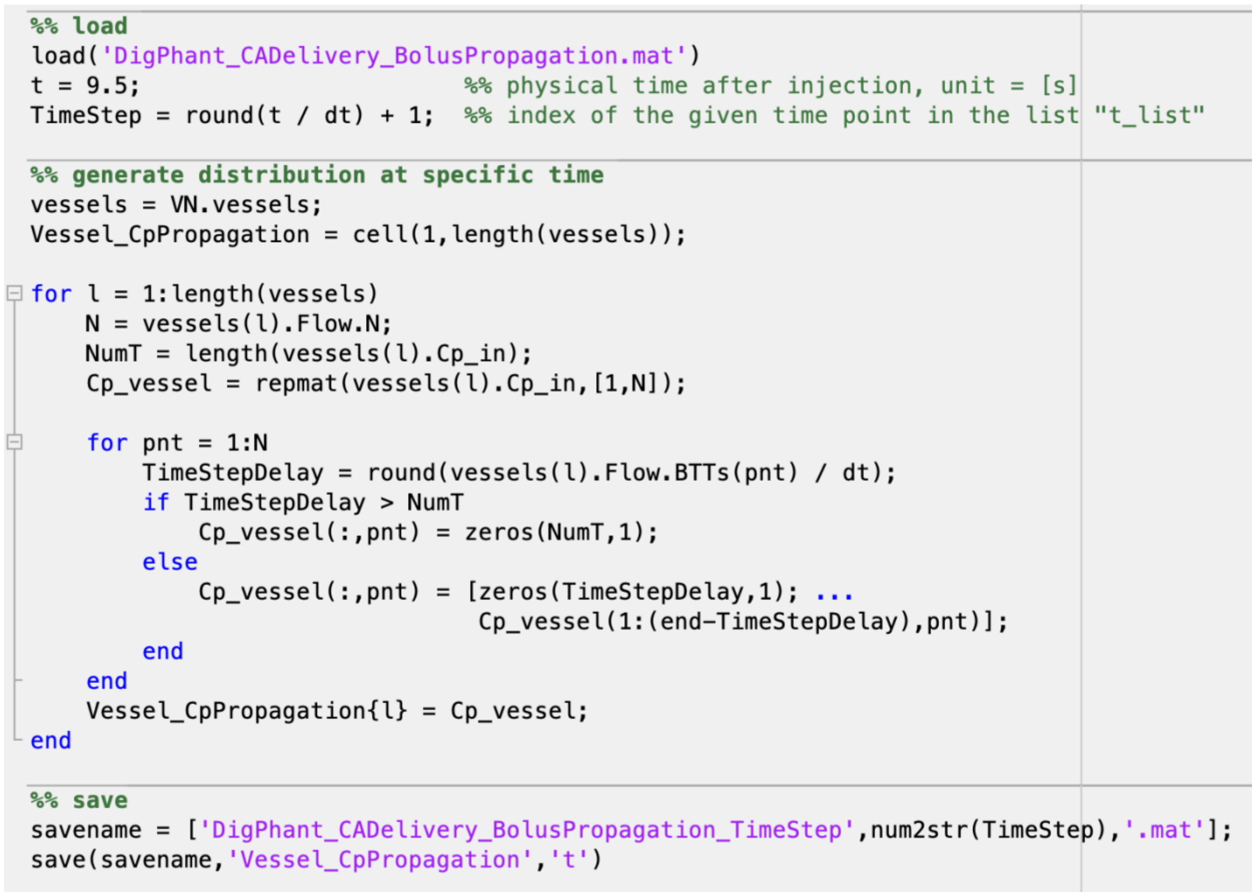
Variable Name	Notes		
dt	Temporal step-width to define vascular bolus concentration profile; = 0.5s		
t_list	Time points the values of bolus concentration are given, with unit = [s]		
VN	Data structure storing information of bolus propagation through the entire vasculature, with involved fields listed below --		
	.dims	matrix size of the phantom grid; = [706, 361, 576]	
	.VM	Binary map indicating the voxels belonging to region of the vasculature.	
	.vessels	Data structure storing information of individual vessel. Size = 1 × #(vessels), for each element represent one vessel (e.g., l^{th}). Involved fields listed below --	
		.n1	Node index of the starting point of this vessel (refers to the element in ‘VN.nodes’)

		.n2	Node index of the ending point of this vessel (refers to the elements in 'VN.nodes')	
		.voxel	Voxel indices of points long this vessel	
		.Flow	Geometry of smoothed vessels and information obtained from solved blood flow are stored in this field. Involved fields list below --	
			.dl	Spatial step-width of solving, with unit of [voxel] = 0.031 mm
			.N	Number of points along this vessel after smoothing
			.X	x-coordinates (i.e., 2nd dimension in the 3D image array) of points along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.Y	y-coordinates (i.e., 1st dimension in the 3D image array) of points along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.Z	z-coordinates (i.e., 3rd dimension in the 3D image array) of points along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.R	Local radius along the smoothed vessel, with unit of [voxel] = 0.031 mm
			.idx	Global indices of points along this smoothed vessel in the whole vascular tree (terminal ends labeled as NaN)
			.drct	Label indicating flow direction in comparison to the direction that the vascular geometry is stored: same as the geometry = 1; opposite as the geometry = -1
			.n_in	Node index of the inlet point of blood flow of this vessel: VN.vessels(*l*).Flow.n_in = VN.vessels(*l*).n1, if VN.vessels(*l*).Flow.drct = 1; VN.vessels(*l*).Flow.n_in = VN.vessels(*l*).n2, if VN.vessels(*l*).Flow.drct = -1.
			.n_out	Node index of the outlet point of blood flow of this vessel; VN.vessels(*l*).Flow.n_out = VN.vessels(*l*).n2, if VN.vessels(*l*).Flow.drct = 1; VN.vessels(*l*).Flow.n_out = VN.vessels(*l*).n1, if VN.vessels(*l*).Flow.drct = -1.
			.up_v	Indices of parent (i.e., inlet) vessels: Vessels directly connected to this vessel on the upstream
			.up_n	Indices of parent nodes: Nodes connected to this vessel by parent vessels.
			.dw_v	Indices of children (i.e., outlet) vessels: Vessels directly connected to this vessel on the downstream
			.dw_n	Indices of children nodes: Nodes connected to this vessel by children vessels.
			.BTTs	Time for bolus traveling from the flow inlet of this vessel to each point along this vessel, with unit = [s]
			.BAT	Bolus-arrival time at the flow inlet of this vessel since the injection of contrast agent, with unit = [s]
		.Cp_in		Bolus concentration profile at the flow inlet of this vessels (i.e., VN.vessels(*l*).n_in), with unit = [mM]

	.nodes	Data structure storing information of each nodes (i.e., branch or terminal points) within the vasculature Size = 1 × #(nodes), for each element represent each node (e.g., n^{th}). Involved fields listed below --		
		.voxel	Voxel index of this node	
		.conn_v	Indices of vessels connected to this node. (refers to the elements in 'VN.vessels')	
		.conn_n	Indices of nodes that this node is connected to. (refers to the elements in 'VN.nodes')	
		.comy	y-coordinate of this node, with unit of [voxel] = 0.031 mm	
		.comx	x-coordinate of this node, with unit of [voxel] = 0.031 mm	
		.comz	z-coordinate of this node, with unit of [voxel] = 0.031 mm	
		.end	Label indicating if the node is a branching or terminal point: branching = 0, terminal = 1	
		.Flow	Information obtained from blood flow solution stored in this field. Involved fields list below --	
			.up_v	Indices of parent (i.e., inlet) vessels: Vessels directly connected to this node on the upstream.
			.up_n	Indices of parent nodes: Nodes connected to this node by parent vessels.
			.dw_v	Indices of children (i.e., outlet) vessels: Vessels directly connected to this node on the downstream.
			.dw_n	Indices of children nodes: Nodes connected to this node by children vessels.
			.initial	Label if this node is an inlet of the entire vasculature: An inlet = 1; not an inlet = 0.
			.BAT	Bolus-arrival time at this node, unit = [s]
			.w_in	Partition(s) of blood flux this node get from each of its parent vessel(s)
			.w_out	Partition(s) of blood flux this node outputting to each of its children vessel(s)
		.Cp	Bolus concentration profile at this node, unit = [mM]	

- ***Loading & Visualization via MATLAB and python scripts (optional):***

With the information contained in this file, we can generate distribution of contrast agent in vasculature at a specific time point. For example, to generate distribution at 9.5 s after injection --



```
%% load
load('DigPhant_CADelivery_BolusPropagation.mat')
t = 9.5;                       %% physical time after injection, unit = [s]
TimeStep = round(t / dt) + 1;  %% index of the given time point in the list "t_list"

%% generate distribution at specific time
vessels = VN.vessels;
Vessel_CpPropagation = cell(1,length(vessels));

for l = 1:length(vessels)
    N = vessels(l).Flow.N;
    NumT = length(vessels(l).Cp_in);
    Cp_vessel = repmat(vessels(l).Cp_in,[1,N]);

    for pnt = 1:N
        TimeStepDelay = round(vessels(l).Flow.BTTs(pnt) / dt);
        if TimeStepDelay > NumT
            Cp_vessel(:,pnt) = zeros(NumT,1);
        else
            Cp_vessel(:,pnt) = [zeros(TimeStepDelay,1); ...
                                Cp_vessel(1:(end-TimeStepDelay),pnt)];
        end
    end
    Vessel_CpPropagation{l} = Cp_vessel;
end

%% save
savename = ['DigPhant_CADelivery_BolusPropagation_TimeStep',num2str(TimeStep),'.mat'];
save(savename,'Vessel_CpPropagation','t')
```

4.2 Subfolder: ‘H5_Tissue’

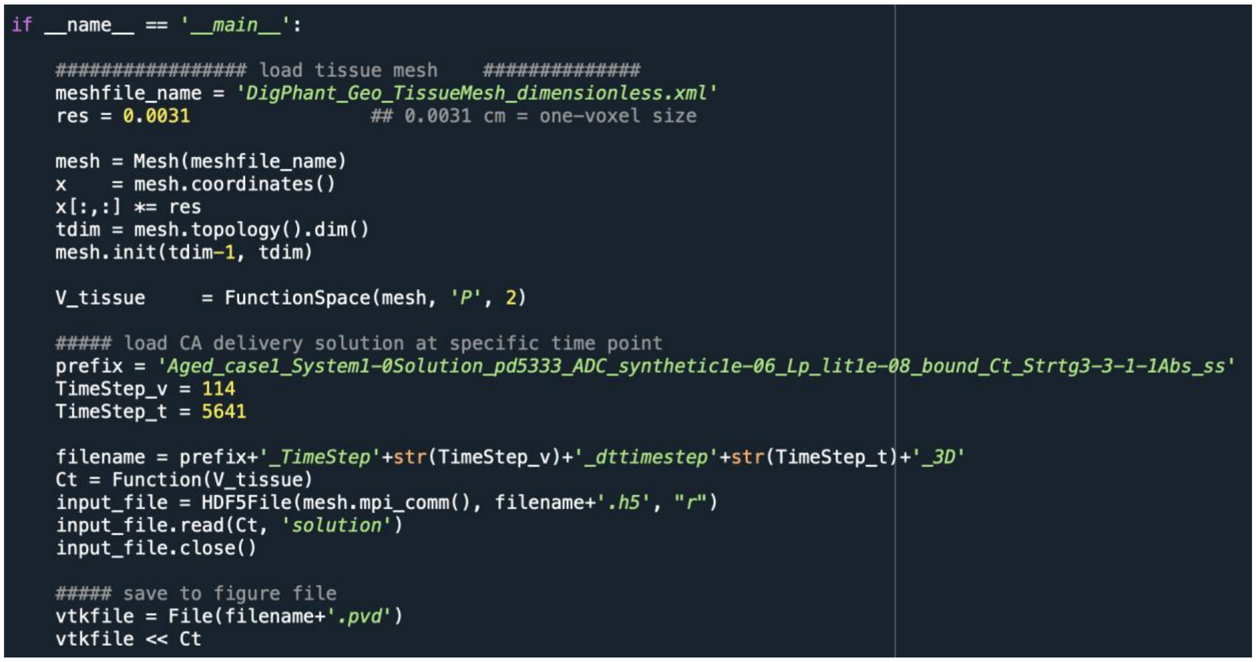
The delivery of contrast agent in the interstitial tissue within one minute after injection is simulated with a temporal step-width of 0.01 s. Every five time steps, a contrast agent distribution solution is saved as a mesh-based data file (i.e., one file in the sub-folder, ‘H5_Tissue’). Therefore, 1200 files are included in this sub-folder, resulting in a temporal resolution of 0.05 s for the generated contrast agent dynamics. Users are encouraged to download part or all of these files according to their own need for temporal resolution.

Each file in this folder is named by the rule of ‘Prefix + Time step for the input vascular bolus profile + Time step of numerical simulation in the interstitial tissue’. Note that bolus concentration is given with a temporal resolution of 0.5 s. For example, the file

‘Aged_case1_System1-0Solution_pd5333_ADC_synthetic1e-06_Lp_lit1e-08_bound_Ct_Strtg3-3-1-1Abs_ss_TimeStep114_dttimestep5641_3D.h5’ means it stores the distribution in the tissue at 5641 × 0.01 = 56.41 s after the injection of contrast agent, where the simulation used the vascular bolus concentration given at (114 – 1) × 0.5 = 56.5 s as the source input.

- ***Loading via python script:***

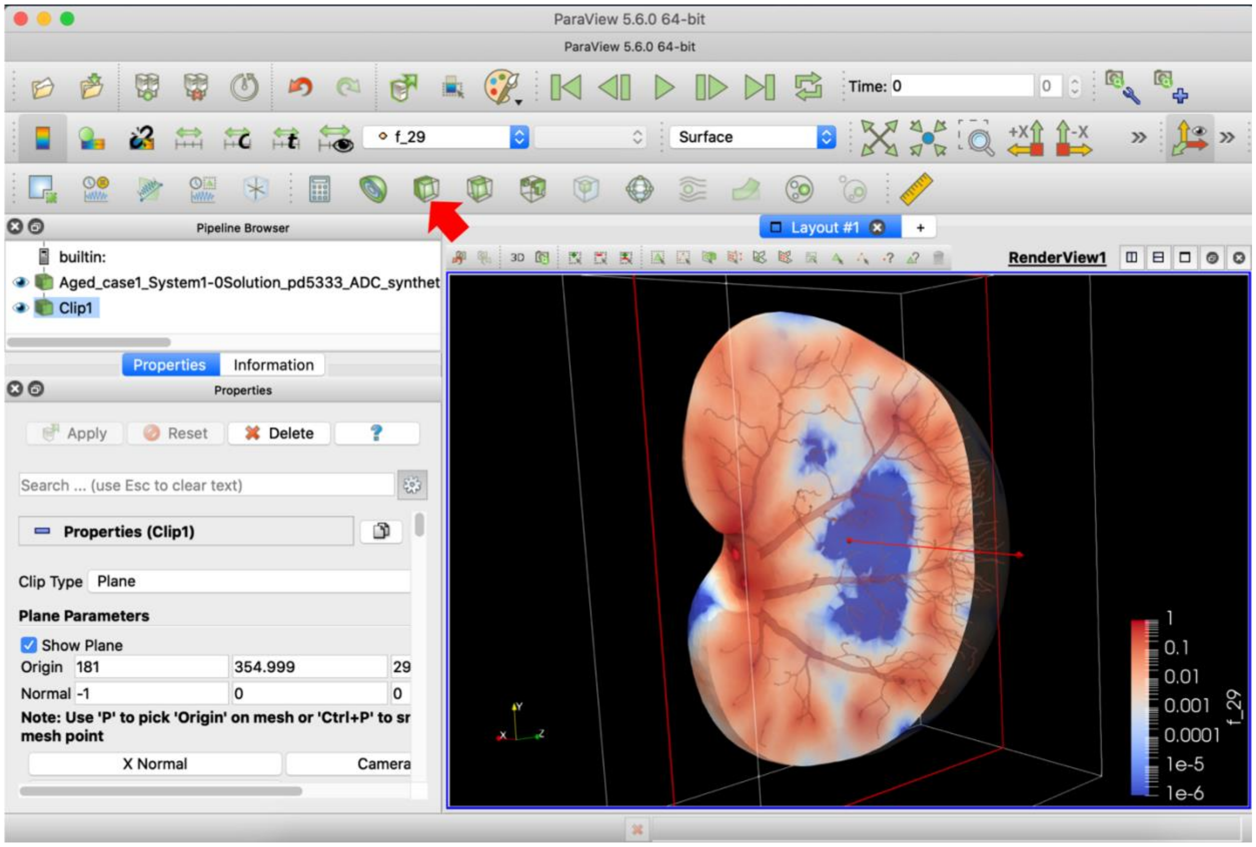
For example, to load the distribution at time 56.41 s –



```
if __name__ == '__main__':

    ################ load tissue mesh    #############
    meshfile_name = 'DigPhant_Geo_TissueMesh_dimensionless.xml'
    res = 0.0031                ## 0.0031 cm = one-voxel size

    mesh = Mesh(meshfile_name)
    x    = mesh.coordinates()
    x[:,:] *= res
    tdim = mesh.topology().dim()
    mesh.init(tdim-1, tdim)

    V_tissue     = FunctionSpace(mesh, 'P', 2)

    ##### load CA delivery solution at specific time point
    prefix = 'Aged_case1_System1-0Solution_pd5333_ADC_synthetic1e-06_Lp_lit1e-08_bound_Ct_Strtg3-3-1-1Abs_ss'
    TimeStep_v = 114
    TimeStep_t = 5641

    filename = prefix+'_TimeStep'+str(TimeStep_v)+'_dttimestep'+str(TimeStep_t)+'_3D'
    Ct = Function(V_tissue)
    input_file = HDF5File(mesh.mpi_comm(), filename+'.h5', "r")
    input_file.read(Ct, 'solution')
    input_file.close()

    ##### save to figure file
    vtkfile = File(filename+'.pvd')
    vtkfile << Ct
```

- ***Visualization via ParaView:***

As shown in Fig 4.1, we open the produced VTK file of mesh-based solution of contrast agent distribution in the interstitial tissue in ParaView. To visualize the distribution within the kidney tissue, we set the property ‘Surface Opacity’ of the original 3D volume to be 0.2 and apply the operator ‘Clip’ (as indicated by the red arrow in Fig 4.1) on the 3D volume to generate a new chopped view.



builtin:
Aged_case1_System1-0Solution_pd5333_ADC_synthet
Clip1

Properties (Clip1)

Clip Type Plane

Plane Parameters

☑ Show Plane

Origin 181 | 354.999 | 29

Normal -1 | 0 | 0

Note: Use 'P' to pick 'Origin' on mesh or 'Ctrl+P' to sr
mesh point

Figure 4.1

Screenshot of visualizing 3D mesh-based solution of contrast agent distribution in ParaView.

4.3 Subfolder: 'Mat_WholeDomain'

Combining the outputs from simulation in tissue and solution of vascular bolus propagation, we can generate the time-dependent distribution of contract agent in the whole domain. For convenience of further usage, the distributions in the whole domain have been converted to voxelized maps and saved as MATLAB-files. In total, 240 files are included in the sub-folder, 'Mat_WholeDomain', resulting in a temporal resolution of 250 ms for the contrast agent dynamics. Again, users are encouraged to download part or all of these files according to their own need for temporal resolution.

Each file in this folder is named by the rule of 'Prefix + Time step of numerical simulation in the interstitial tissue'.

For example, the file 'Aged_case1_System1-0Solution_pd5333_ADC_synthetic1e-06_Lp_lit1e-08_bound_TotalConcentration_dttimestep_1101.mat' means it stores the voxel-wised distribution in the whole domain at $1101 \times 0.01 = 11.01$ s after injection of contrast agent.

- ***Variables:***

Variable Name	Type	Notes
TotalConcentration_fill	double	Voxel-wised concentration of contrast agent, unit = [mM]

- ***Loading & Visualization via MATLAB script:***

```
close all
Prefix = 'Aged_case1_System1-0Solution_pd5333_ADC_synthetic1e-06_Lp_lit1e-08_bound_TotalConcentration_dttimestep_';
TimeStep_tissue = 5451;
C_filename = [Prefix, num2str(TimeStep_tissue),'.mat'];

load(C_filename)
load('DigPhant_Geo_FinalMasks.mat')

Mask_whole = double(Masks);
Mask_whole(Mask_whole == 0) = NaN;
C_whole = TotalConcentration_fill .* Mask_whole;

c = jet;
c(1,:) = [0,0,0];

figure
imagesc(log(C_whole(:,:,288)))
axis equal tight off
colormap(c)
colorbar
ax = gca(gcf); ax.FontSize = 16;
```

- ***Visualization Outputs:***

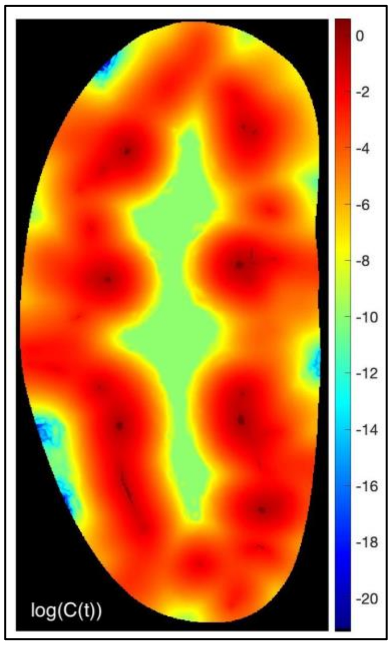


Figure 4.2

Contrast agent distribution (C [mM]) map in the whole kidney tissue at 51.51 s after injection (visualized on the central slice). The color-bar is converted to log scale for better contrast.