Cardiovascular Engineering

CVEAME

0-D/3-D Multiscale Simulations in FLUENT

contact:

michael.neidlin@gmail.com

Dr. Michael Neidlin

Cardiovascular Engineering (CVE)

RWTH AACHEN UNIVERSITY

What is this presentation about?

- This presentation provides an extensive explanation on how to manage multiscale (0-D/3-D) CFD models with ANSYS Fluent.
- The application is vascular flow in the aortic branch with a coupling to a 0-D (lumped parameter network) of the entire cardiovascular system.
- The scripts developed were applied for simulations of hemodynamics during various outflow grafting positions under LVAD support. If you use this code, please cite:

Neidlin et al. 2016, doi: 10.1016/j.jbiomech.2016.06.003

The general methodology was developed by Migliavacca et al, so please acknowledge accordingly:

Migliavacca et al. 2006 doi: 10.1016/j.jbiomech.2005.02.021

Why do we need multiscale simulations?

- (Almost) all numerical models are embedded in a dynamical system. If we look at one part of the system in 3-D we need to take the remaining system into consideration
- Our premise is that our flow solution variables such as pressure and flow taken from 3-D influence the lower scale system (0-D).
- Changes in 0-D domain may have an influence (e.g. boundary cond.) on 3-D field -> Two way coupling
- If there are no changes, (e.g. cellular processes which do not affect the flow feeld) -> one way coupling. Only 3-D influences 0-D

Aim

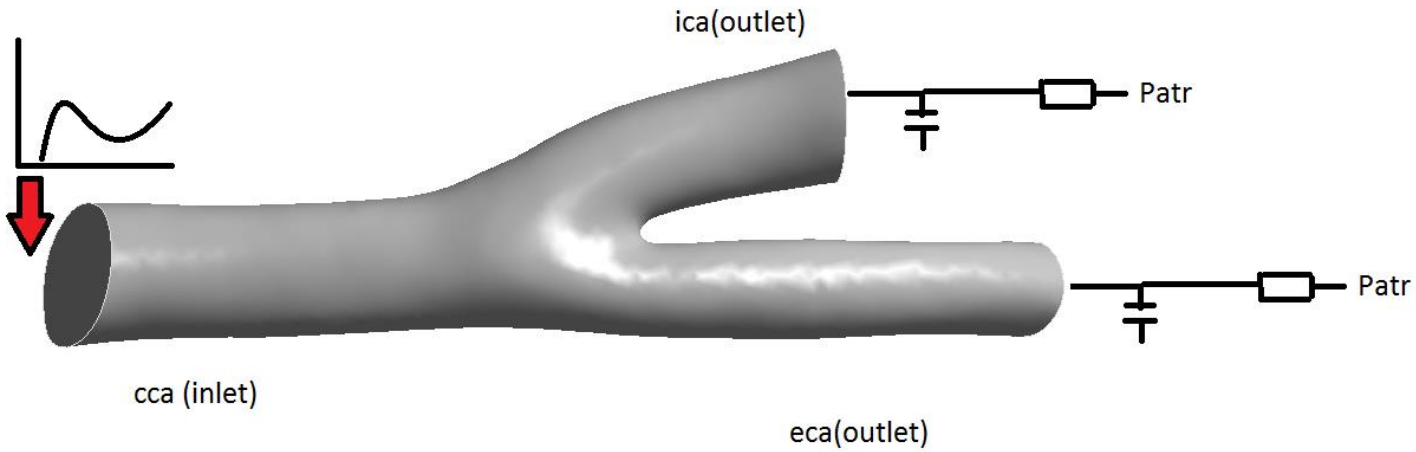
- Coupling of Lumped Parameter models to 3-D CFD (or FSI) simulations to get appropriate b.c., look at publications from Francesco Migliavacca/ Giancarlo Pennati regarding multiscale/CFD simulations (2004-2006)!
- Necessary in e.g. vessel flows
- What do you need to know in advance
  - What are lumped parameter models
  - How to use **compiled (!!!) UDF**
    - Look for tutorials delivered by ANSYS, Sometimes PATH variable (system variable) has to be changed in Win 7 and Microsoft Visual Studio 2012 (or higher) needs to be installed. There are a lot of explanations on the web on how to get compiled UDF and Win7 working
  - How to create **and** solve lumped parameter models, not only in Simulink but also in C (e.g. MATLAB). You need to have a working C code (MATLAB) code of your ODE system before you can proceed with the coupling. Sample Codes for Simulink and Matlab are included

Structure

- Explanation of MS simulation with an easy file
- Further explanation and sample WB file of more difficult case
- Tips&Tricks

Easy example

- An easy UDF represents this case to explain general structure of code



- Example is: carotid artery with pulsatile inflow and 2 R-C models at the outlets, Patr is the atrial pressure = 30 mmHg
- Go through the following slides, can also look at UDF, The Fluent file is also given. Multiscale\Carotid_MS\UDF\Carotid_MS.c

Explanation of UDF I

```
#define nSV 2/*# state variables*/
#define DENSITY 1056.4
real Qeca,Qica;      /*flows from LPN*/
real QFeca,QFica; /*flows calculated by Fluent*/
real Peca,Pica;    /*Pressures from LPN*/
real T,DT,TP1,TP2,INT,STP;

int i,j,NA,NSTEP;


real X[nSV],X1[nSV],X2[nSV]; /*Pressure arrays*/
real DX[nSV],FF[nSV]; /*Delta P and Flow arrays*/
real R[nSV],C[nSV]; /*Resistance and compliance*/


DEFINE_INIT(initialization,domain)
/*remember to put init file (with TP1=TP2=0) and solnet file (with the same left and right columns) in the sim folder*/
{
    #if !RP_NODE    /*SERIAL or HOST*/
        FILE *init,*parameters,;


        if ((init=fopen("init.txt","r"))==NULL)
            Message("\nError reading file init.txt\n");
        else
        {
            init=fopen("init.txt","r");
            fscanf(init,"%lf %lf \n %lf %lf %lf  \n",&TP1, &TP2, &INT, &Peca,&Pica);
            fclose(init);
            NSTEP=0;
            Message("initialization \n%lf %lf %lf %lf %lf \n",TP1,TP2,INT,Peca,Pica);
        }
```

Approach is following: In each timestep Fluent calculates flows across Boundaries (QFeca/QFica). These values are converted to ml/s (Qeca,Qica) and are taken as input parameters for LP model. LP model solves ODEs and applies computed pressures as boundary pressure in Fluent

X are pressures for ODE system, FF are flows for ODE system. [i] for each compartment

R and C are the according resistances and compliances

#if !RP_NODE (and everything which is brown colour) is needed if parallel computing is happening

Initial time (TP1/TP2)=0 INT=Timestep size and intital Pressures are read out of txt

Explanation of UDF II

```
/*FROM HERE OWN CODE FOR OPEN LOOP CIRCULATION*/

/*DECLARATION OF PARAMETERS*/
        if ((parameters=fopen("parameters.txt","r"))==NULL)
            Message("\nError reading file parameters.txt\n");
        else
        {
            parameters=fopen("parameters.txt","r");
            for (i=0;i<nSV;i++)
            fscanf(parameters,"%lf %lf \n",&R[i], &C[i]);
            fclose(parameters);

            Message("\n First column: Resistances, Second: Capacitances \n");
            for (i=0;i<nSV;i++)
            Message("%lf %lf\n",R[i], C[i]);
        }



        X2[0]=10; /*Intital pressures eca and ica*/
        X2[1]=10;


#endif
```

Intialisation of pressures and R and C arrays, easy stuff self explanatory. the #endif in the end ends the !RP_NODE loop

Explanation of UDF III

```
DEFINE_ADJUST(BOUNDARY,domain)
{
    #if !RP_NODE    /*SERIAL or HOST*/
        int i,FLAG;
        real Patr;

    #endif

    node_to_host_real_2(QFeca,QFica);

    #if !RP_NODE    /*SERIAL or HOST*/
        T = RP_Get_Real("flow-time");
        Message("T: %lf\n",T);
        Patr=30; /*mmHg*/

    /*SOLUTION OF ODES WITH REAL TIME*/


        if (fabs(T-TP2)<=0.000001)  /*the LPN stops while the 3D model (Fluent) is iterating in the same time step*/
        {
            DT=INT;
            FLAG=2;
        }
        else if ((T-TP2)>0.000001)  /*jump to the next time step as Fluent has converged*/
        {
            DT=(T-TP2);
            INT=DT;
            TP1=TP2;
            TP2=T;
            FLAG=1;
            NSTEP=NSTEP+1;
        }
        else    /*time step is decreased as Fluent has not converged yet*/
        {
            DT=T-TP1;
            INT=DT;
            TP2=T;
            FLAG=0;
        }

        Message("FLAG:%d\n",FLAG);

        if (FLAG!=2)
        {
            Qeca=(QFeca/DENSITY)*pow(10,6);
            Qica=(QFica/DENSITY)*pow(10,6);
```

node_to_host necessary during parallel computation. „Gives“ the measured mass flow rates to the LPN network.
important: the maximum number is 7, if you have more than 7 boundaries than you have to add another line: node_to_host_real_7 and node_to_host_real_2 (in case of 9 boundaries(

This whole block is necessary to check whether
LPN solution converged, if not then the LPN timestep is decreased.
This is very unlikely, so (almost) always the case FLAG=1 is happening

Transfer from measured mass flow rate (FLUENT) to ml/s

Explanation of UDF IV

```
        if (FLAG==1)    /*go to next time-step*/
            for (i=0; i<nSV; i++)
            {
                X[i]=X2[i];
            }
        else        /*decrease time-step*/
            for (i=0; i<nSV; i++)
            {
                X[i]=X1[i];
            }
    /*RC Equations for ECA*/

    FF[0]=(X[0]-Patr)/R[0];
    DX[0]=(Qeca-FF[0])/(C[0]);

    /*RC Equations for ICA*/

    FF[1]=(X[1]-Patr)/R[1];
    DX[1]=(Qica-FF[1])/(C[1]);


/*Explicit Euler for ODE*/
    X[0]=X[0]+DT*DX[0];
    X[1]=X[1]+DT*DX[1];


    if (FLAG==1)
    for (i=0; i<nSV; i++)
    {
    X1[i]=X2[i];
    X2[i]=X[i];


    }
    else
    for (i=0; i<nSV; i++)
    {
    X2[i]=X[i];


    }


    Peca=(X[0])*133.32;
    Pica=(X[1])*133.32;


    }   /*end if(FLAG!=2)*/


    #endif  /*!RP_NODE*/

    host_to_node_real_2(Peca,Pica);
```

These are the ODEs: Q=(Pin-Pout)/R and dPin/dt=(Qin-Qout)/C

Explicit euler, here other solutions schemes can be implemented (4 RK or (this one is better) **implicit backward euler!)**

Transfer from mmHg to Pa, Peca and Pica will be applied at boundaries

Explanation of UDF V

```
DEFINE_PROFILE(outlet_eca,thread,nv) /*Pressure BCs, read Fluent MF apply LPN pressure*/
{
    face_t f=0;
    real QF_par=0.0;

    #if !RP_HOST    /*SERIAL or NODE*/
        begin_f_loop(f,thread)
            if(PRINCIPAL_FACE_P(f,thread))
            {
                QF_par += (F_FLUX(f,thread));

                F_PROFILE(f,thread,nv) = Peca;

            }
        end_f_loop(f,thread)
    #endif  /*!RP_HOST*/

    QFeca=(PRF_GRSUM1(QF_par));
}
```

This is the actual coupling. QF_par+=… reads the mass flow from CFD (+= for outlet, -= for inlets)

here the pressure from LPN is applied

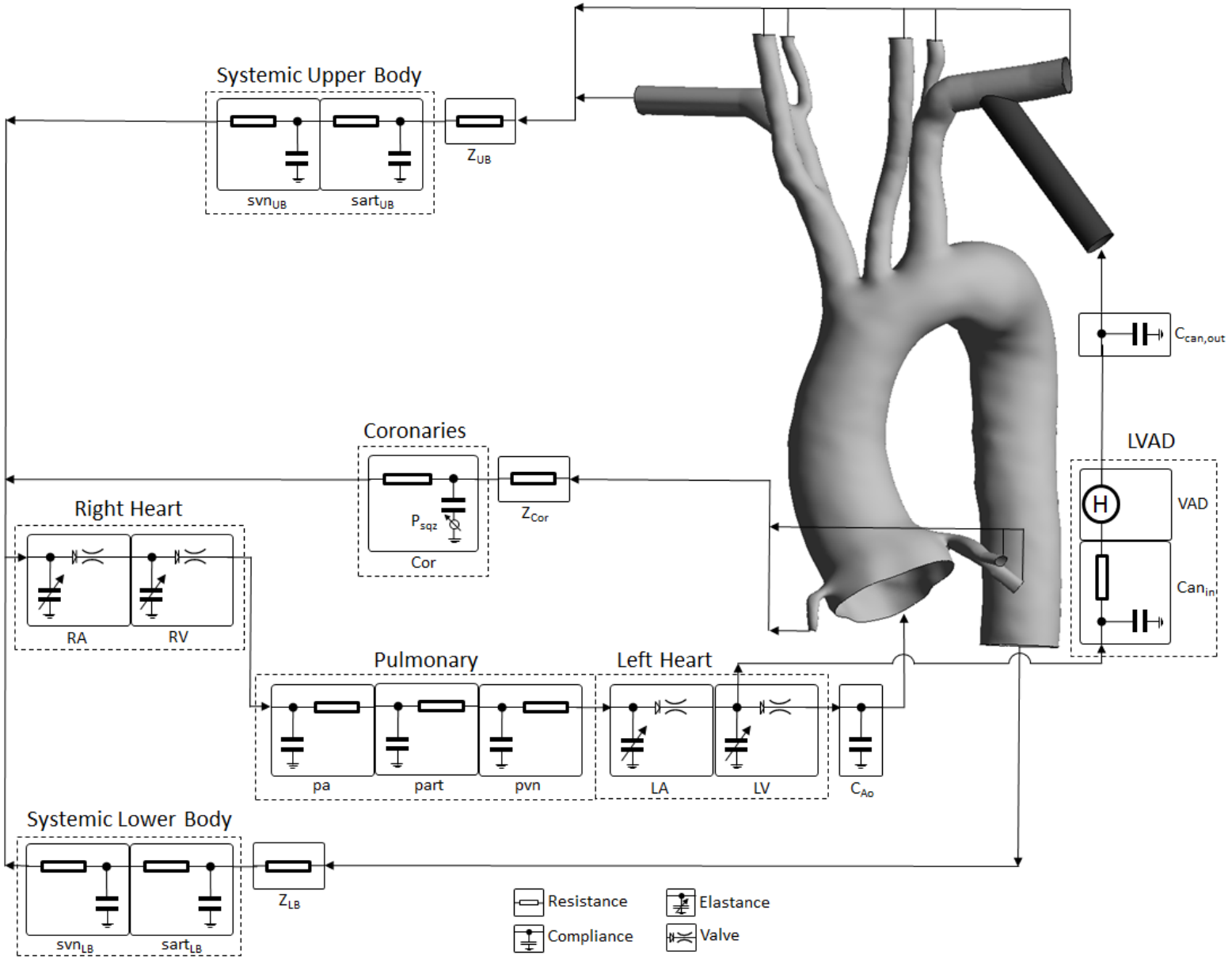
Sums over all nodes and calculated the MF

```
DEFINE_PROFILE(outlet_ica,thread,nv) /*Pressure BCs, read Fluent MF apply LPN pressure*/
{
    face_t f=0;
    real QF_par=0.0;

    #if !RP_HOST    /*SERIAL or NODE*/
        begin_f_loop(f,thread)
            if(PRINCIPAL_FACE_P(f,thread))
            {
                QF_par += (F_FLUX(f,thread));

                F_PROFILE(f,thread,nv) = Pica;

            }
        end_f_loop(f,thread)
    #endif  /*!RP_HOST*/

    QFica=(PRF_GRSUM1(QF_par));
}
```

Step-by-step example

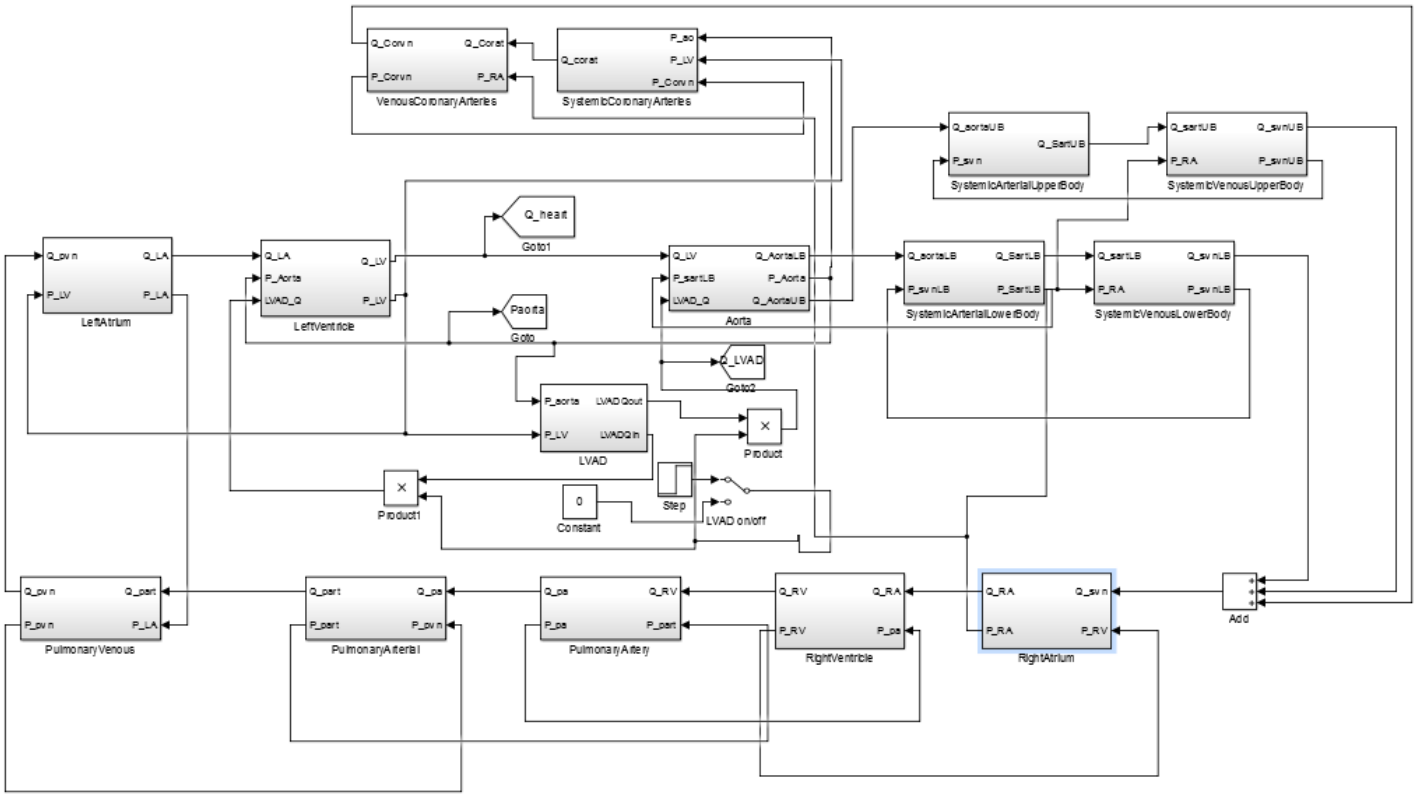
- Now let‘s make things more interesting and look at a 3-D aortic arch with VAD which is included in the entire cardiovascular system with a running pump
- You will see following things:
- 1. Schematic structure
- 2. How does it look like in Simulink or Matlab?
- 3. How to couple it to Fluent?

Schematic structure (Neidlin2016, J Biomech)



Simulink and Matlab

- **LPN\Simulink\LPN_CVS_LVAD.slx**



Matlab

- **LPN\Matlab\cvs_cor_lvad_150408.m**

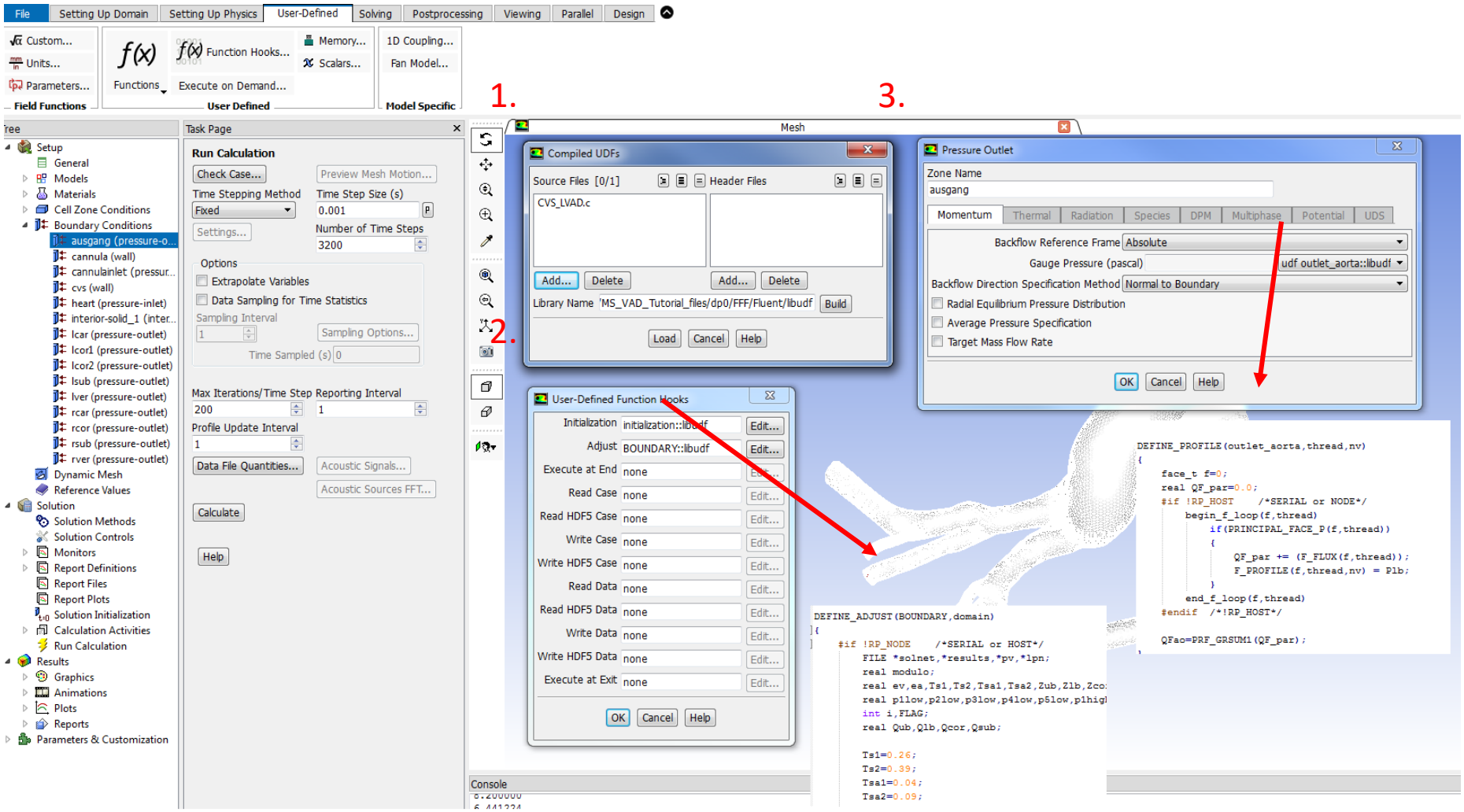
```
%Equations for CoronaryArteries
FF(13)=(X(13)+0.75*PP(11)-PP(5))/(R(13)); %!!!
                                          %C
                                          %D
DX(13)=(FF(12)-FF(3)-FF(1)-FF(13))/C(13);

%Equations for SartUB
 FF(1)=(X(1)-X(2))/R(1);
 DX(1)=(FF(12)-FF(3)-FF(13)-FF(1))/(C(1));


%Equations for SvnUB
FF(2)=(X(2)-PP(5))/R(2);
DX(2)=(FF(1)-FF(2))/C(2);

%Equations for SartLB
FF(3)=(X(3)-X(4))/R(3);
DX(3)=(FF(12)-FF(13)-FF(1)-FF(3))/(C(3));

%Equations for SvnLB
FF(4)=(X(4)-PP(5))/R(4);
DX(4)=(FF(3)-FF(4))/C(4);
```

- Matlab describes the same as Simulink and is transferred 1 to 1 in Fluent UDF with some change in syntax. The Simulink model is a simplified LP model of CVS (only R and C). One can also look at LPN models. However this model is recommended because of its simplicity

Coupling to FLUENT, example DescendingAorta NO LVAD, just CVS + coronaries

Multiscale\OwnCVS_UDF only\CVS_coronary.c

1. Compile UDF, 2. Hook the according functions, 3. Add as pressure b.c



Tips&Tricks

- ALWAYS (!) model the LPN in MATLAB (or C) first and check if the ODE system is working correctly, also use this code to identify parameters R and C
- Write out the results (Pressures /flows) from Fluent into txt to check if there are problems
- More complicated LPNs can cause divergence in FLUENT because INITIAL pressures are poorly chosen (too big differences in pressures =>big mass flows in fluent => bigger differences in pressures), so monitor the X[i] and see if this is the cause
- Damping resistances (RCR instead CR elements can prevent oscillations in flow (shown on slide 18)
- ALWAYS take flow from CFD and give back pressure from LPN! This fact needs a trick if LPN is upstream of CFD (coupling to aortic sinus, see slide 17)
- There is one complete LP UDF for CVS+Coronaries+VAD (CF_VAD.c), which is working, however it is STRONGLY suggested to create OWN LP UDFs!!!
- All credit goes to LaBS in Milano, if lost check out the website. http://www.labsmech.polimi.it/ Pennati&Migliavacca did a lot of this probably very willing to help.

Upstream coupling

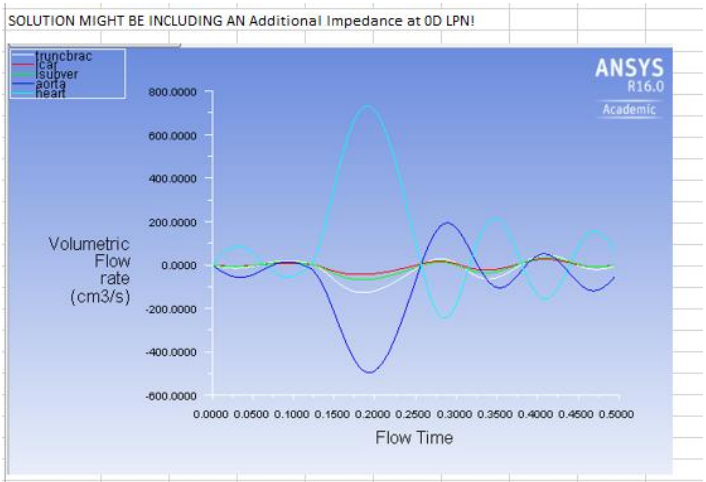
- Taking Q from CFD and give back P from LPN is straightforward if LPN is downstream of CFD. In the other case (if you have closed loop), one has to add an additional compliance (ONLY C) element between LPN and 3-D model to allow the same structure Q_cfd -> P_lpn

```
/*Equations for LV*/
  if ((PP[11]-X[0])>0.0001)
  FF[11]=sqrt(PP[11]-X[0])*R[11];
  else
  FF[11]=0;

PP[11]=(V[11])*ev*EmaxLV+(1-ev)*(LV_Pd_alpha*exp(LV_Pd_kappa*(V[11]))+LV_Pd_beta); /*PRESSURE From Elastance*/
DV[11]=(FF[10]-FF[11]-FF[13]);

/*Equations for AorticSinusCompliace*/
DX[0]=(FF[11]-Qheart)/0.2;



Pheart=(X[0])*133.32;
```

Without X[0], one would have to include Pheart in equation for LV! 0.1-0.3 is a good value for compliance of aortic sinus

Flow oscillations at boundaries

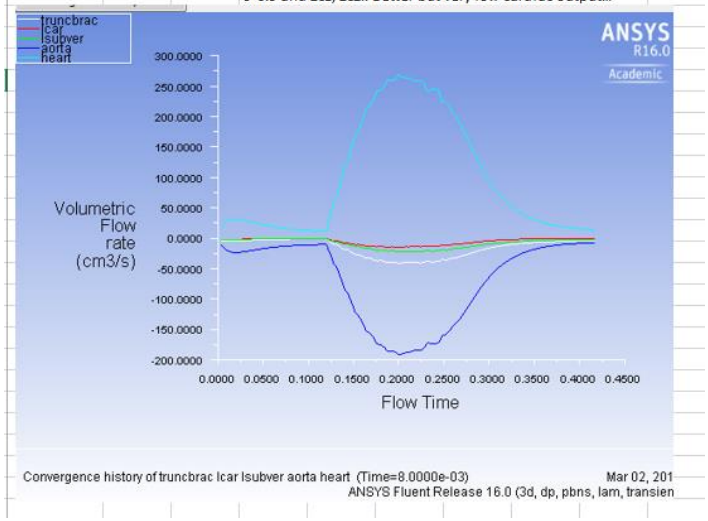
- Addition of damping resistances (small resistance 0.05-0.1 at 3-D/0-D interface). This has the following effect:



Without damping (characteristic impedance)

Flow oscillations are seen!

C=0.3 and Zc1, Zc2.. Better but very low cardiac output...



With damping

```
Pub=(X[1]+Zub*Qub)*133.32;
Psub=(X[1]+Zsub*Qsub)*133.32;
Plb=(X[3]+Zlb*Qlb)*133.32;
Pcor=(X[12]+Zcor*Qcor)*133.32;
Pheart=(X[0])*133.32;
Pcan=(X[14]-0.5*Qcan)*133.32;
```