FEM Tools Workflow

As of March 2021

Created by Anna Knight

FEM Tools Workflow

5. To make your own Field II Loads instead of defaults

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

FEM Tools Workflow

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

2. Launch a bash script

```
[aek27@dcc-login-01  /work/aek27/TestingFemTools2021/TI_attempt3 $ cat TIsimBashLauncher.sh
#!/bin/bash
#SBATCH -o /work/aek27/TestingFemTools2021/SlurmOuts/slurm.%A.%a.out        # STDOUT
#SBATCH -e/work/aek27/TestingFemTools2021/SlurmOuts/slurm.%A.%a.out        # STDERR
#SBATCH --mem=64G
#SBATCH --partition=ultrasound
#SBATCH --exclude=dcc-ultrasound-01
#SBATCH --cpus-per-task=12


date
hostname

module load Python/3.8.1
source /work/aek27/.venv/bin/activate

python runTIsim.py
```

FEM Tools Workflow

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

2. Launch a bash script

```
[aek27@dcc-login-01  /work/aek27/Testing
#!/bin/bash
#SBATCH -o /work/aek27/TestingFemTools2
#SBATCH -e/work/aek27/TestingFemTools20
#SBATCH --mem=64G
#SBATCH --partition=ultrasound
#SBATCH --exclude=dcc-ultrasound-01
#SBATCH --cpus-per-task=12


date
hostname

module load Python/3.8.1
source /work/aek27/.venv/bin/activate

python runTIsim.py
```

3. Standard Python script to make nodes, elems, bc, pmls, etc.

```
[aek27@dcc-login-01  /work/aek27/TestingFemTools2021/TI_attempt3 $ cat runTIsim.py
from os import environ, system
from socket import gethostname
from time import ctime

from fem.mesh import GenMesh, bc
from fem.mesh.GaussExc import generate_loads
from fem.post.create_disp_dat import create_dat as create_disp_dat
from fem.post import create_res_sim

print('STARTED: {}'.format(ctime()))
print('HOST: {}'.format(gethostname()))

##DYNADECK = 'gauss_qsym_pml.dyn'
DYNADECK = 'ortho_tilt11_rot0.dyn'
NTASKS = environ.get('SLURM_NTASKS', '8')

GenMesh.run((-2.0, 2.0, -2.0, 2.0, -4.0, 0.0), (160, 160, 160))

# setup no-symmetry condition
pml_elems = ((5, 5), (5, 5), (5, 5))
face_constraints = (('1,0,0,0,1,1', '1,0,0,0,1,1'),
                    ('0,1,0,1,0,1', '0,1,0,1,0,1'),
                    ('1,1,1,1,1,1', '1,1,1,1,1,1'))
edge_constraints = (((0, 1), (1, 0), (0, 0)), '1,1,0,1,1,1')
bc.apply_pml(pml_elems, face_constraints, edge_constraints)

generate_loads([0.25, 0.25, 0.75], [0.0, 0.0, -1.5])

##system('ls-dyna-d ncpu={} i={}'.format(NTASKS, DYNADECK))
system('singularity exec -p -B /work/aek27/TestingFemTools2021/TI_attempt3 /opt/apps/staging/ls-
dyna-singularity/ls-dyna.sif ls-dyna-d ncpu={} i={} memory=600000000'.format(NTASKS, DYNADECK))



create_disp_dat()

create_res_sim.run(DYNADECK,
                   dispout="disp.dat",
                   ressim="res_sim.mat")
create_res_sim.run(dynadeck=DYNADECK,
                   dispout="disp.dat",
                   ressim='res_sim.h5')
create_res_sim.extract3Darfidata(dynadeck=DYNADECK,
                                 dispout="disp.dat",
                                 ressim="res_sim.pvd")

##os.system("xz -v disp.dat")

print('FINISHED: {}'.format(ctime()))
```

FEM Tools Workflow

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

2. Launch a bash script

```
[aek27@dcc-login-01  /work/aek27/Testing
#!/bin/bash
#SBATCH -o /work/aek27/TestingFemTools2
#SBATCH -e/work/aek27/TestingFemTools20
#SBATCH --mem=64G
#SBATCH --partition=ultrasound
#SBATCH --exclude=dcc-ultrasound-01
#SBATCH --cpus-per-task=12


date
hostname

module load Python/3.8.1
source /work/aek27/.venv/bin/activate

python runTIsim.py
```

3. Standard Python script to make nodes, elems, bc, pmls, etc.

```
[aek27@dcc-login-01  /work/aek27/TestingFemTools2021/TI_attempt3 $ cat runTIsim.py
from os import environ, system
from socket import gethostname
from time import ctime

from fem.mesh import GenMesh, bc
from fem.mesh.GaussExc import generate_loads
from fem.post.create_disp_dat import create_dat as create_disp_dat
from fem.post import create_res_sim

print('STARTED: {}'.format(ctime()))
print('HOST: {}'.format(gethostname()))

##DYNADECK = 'gauss_qsym_pml.dyn'
DYNADECK = 'ortho_tilt11_rot0.dyn'
NTASKS = environ.get('SLURM_NTASKS', '8')

GenMesh.run((-2.0, 2.0, -2.0, 2.0, -4.0, 0.0), (160, 160, 160))

# setup no-symmetry condition
pml_elems = ((5, 5), (5, 5), (5, 5))
face_constraints = (('1,0,0,0,1,1', '1,0,0,0,1,1'),
                    ('0,1,0,1,0,1', '0,1,0,1,0,1'),
                    ('1,1,1,1,1,1', '1,1,1,1,1,1'))
edge_constraints = (((0, 1), (1, 0), (0, 0)), '1,1,0,1,1,1')
bc.apply_pml(pml_elems, face_constraints, edge_constraints)

generate_loads([0.25, 0.25, 0.75], [0.0, 0.0, -1.5])

##system('ls-dyna-d ncpu={} i={}'.format(NTASKS, DYNADECK))
system('singularity exec -p -B /work/aek27/TestingFemTools2021/TI_attempt3 /opt/apps/staging/ls-
dyna-singularity/ls-dyna.sif ls-dyna-d ncpu={} i={} memory=600000000'.format(NTASKS, DYNADECK))



create_disp_dat()

create_res_sim.run(DYNADECK,
                   dispout="disp.dat",
                   ressim="res_sim.mat")
create_res_sim.run(dynadeck=DYNADECK,
                   dispout="disp.dat",
                   ressim='res_sim.h5')
create_res_sim.extract3Darfidata(dynadeck=DYNADECK,
                                 dispout="disp.dat",
                                 ressim="res_sim.pvd")

##os.system("xz -v disp.dat")

print('FINISHED: {}'.format(ctime()))
```

4. Make sure pointing at singularity container created by Duke OIT to use LS-DYNA license. Only need to change folder name (TI_attempt3 here)

FEM Tools Workflow

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

2. Launch a bash script

```
[aek27@dcc-login-01  /work/aek27/Testing
#!/bin/bash
#SBATCH -o /work/aek27/TestingFemTools2
#SBATCH -e/work/aek27/TestingFemTools20
#SBATCH --mem=64G
#SBATCH --partition=ultrasound
#SBATCH --exclude=dcc-ultrasound-01
#SBATCH --cpus-per-task=12


date
hostname

module load Python/3.8.1
source /work/aek27/.venv/bin/activate

python runTIsim.py
```

3. Standard Python script to make nodes, elems, bc, pmls, etc.

```
[aek27@dcc-login-01  /work/aek27/TestingFemTools2021/TI_attempt3 $ cat runTIsim.py
from os import environ, system
from socket import gethostname
from time import ctime

from fem.mesh import GenMesh, bc
from fem.mesh.GaussExc import generate_loads
from fem.post.create_disp_dat import create_dat as create_disp_dat
from fem.post import create_res_sim

print('STARTED: {}'.format(ctime()))
print('HOST: {}'.format(gethostname()))

##DYNADECK = 'gauss_qsym_pml.dyn'
DYNADECK = 'ortho_tilt11_rot0.dyn'
NTASKS = environ.get('SLURM_NTASKS', '8')

GenMesh.run((-2.0, 2.0, -2.0, 2.0, -4.0, 0.0), (160, 160, 160))

# setup no-symmetry condition
pml_elems = ((5, 5), (5, 5), (5, 5))
face_constraints = (('1,0,0,0,1,1', '1,0,0,0,1,1'),
                    ('0,1,0,1,0,1', '0,1,0,1,0,1'),
                    ('1,1,1,1,1,1', '1,1,1,1,1,1'))
edge_constraints = (((0, 1), (1, 0), (0, 0)), '1,1,0,1,1,1')
bc.apply_pml(pml_elems, face_constraints, edge_constraints)

generate_loads([0.25, 0.25, 0.75], [0.0, 0.0, -1.5])

##system('ls-dyna-d ncpu={} i={}'.format(NTASKS, DYNADECK))
system('singularity exec -p -B /work/aek27/TestingFemTools2021/TI_attempt3 /opt/apps/staging/ls-
dyna-singularity/ls-dyna.sif ls-dyna-d ncpu={} i={} memory=600000000'.format(NTASKS, DYNADECK))



create_disp_dat()

create_res_sim.run(DYNADECK,
                   dispout="disp.dat",
                   ressim="res_sim.mat")
create_res_sim.run(dynadeck=DYNADECK,
                   dispout="disp.dat",
                   ressim='res_sim.h5')
create_res_sim.extract3Darfidata(dynadeck=DYNADECK,
                                 dispout="disp.dat",
                                 ressim="res_sim.pvd")

##os.system("xz -v disp.dat")

print('FINISHED: {}'.format(ctime()))
```

4. Make sure pointing at singularity container created by Duke OIT to use LS-DYNA license. Only need to change folder name (TI_attempt3 here)

5. To make your own Field II Loads instead of defaults

```
field2dyna('nodes.dyn', 'field_params_L74_F0p52_focus20mm.json', 'elems.dyn')
```

FEM Tools Workflow

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

2. Launch a bash script

```
[aek27@dcc-login-01  /work/aek27/Testing
#!/bin/bash
#SBATCH -o /work/aek27/TestingFemTools2
#SBATCH -e/work/aek27/TestingFemTools20
#SBATCH --mem=64G
#SBATCH --partition=ultrasound
#SBATCH --exclude=dcc-ultrasound-01
#SBATCH --cpus-per-task=12


date
hostname

module load Python/3.8.1
source /work/aek27/.venv/bin/activate

python runTIsim.py
```

3. Standard Python script to make nodes, elems, bc, pmls, etc.

```
[aek27@dcc-login-01  /work/aek27/TestingFemTools2021/TI_attempt3 $ cat runTIsim.py
from os import environ, system
from socket import gethostname
from time import ctime

from fem.mesh import GenMesh, bc
from fem.mesh.GaussExc import generate_loads
from fem.post.create_disp_dat import create_dat as create_disp_dat
from fem.post import create_res_sim

print('STARTED: {}'.format(ctime()))
print('HOST: {}'.format(gethostname()))

##DYNADECK = 'gauss_qsym_pml.dyn'
DYNADECK = 'ortho_tilt11_rot0.dyn'
NTASKS = environ.get('SLURM_NTASKS', '8')

GenMesh.run((-2.0, 2.0, -2.0, 2.0, -4.0, 0.0), (160, 160, 160))

# setup no-symmetry condition
pml_elems = ((5, 5), (5, 5), (5, 5))
face_constraints = (('1,0,0,0,1,1', '1,0,0,0,1,1'),
                    ('0,1,0,1,0,1', '0,1,0,1,0,1'),
                    ('1,1,1,1,1,1', '1,1,1,1,1,1'))
edge_constraints = (((0, 1), (1, 0), (0, 0)), '1,1,0,1,1,1')
bc.apply_pml(pml_elems, face_constraints, edge_constraints)

generate_loads([0.25, 0.25, 0.75], [0.0, 0.0, -1.5])

##system('ls-dyna-d ncpu={} i={}'.format(NTASKS, DYNADECK))
system('singularity exec -p -B /work/aek27/TestingFemTools2021/TI_attempt3 /opt/apps/staging/ls-
dyna-singularity/ls-dyna.sif ls-dyna-d ncpu={} i={} memory=600000000'.format(NTASKS, DYNADECK))


create_disp_dat()

create_res_sim.run(DYNADECK,
                   dispout="disp.dat",
                   ressim="res_sim.mat")
create_res_sim.run(dynadeck=DYNADECK,
                   dispout="disp.dat",
                   ressim='res_sim.h5')
create_res_sim.extract3Darfidata(dynadeck=DYNADECK,
                                 dispout="disp.dat",
                                 ressim="res_sim.pvd")

##os.system("xz -v disp.dat")

print('FINISHED: {}'.format(ctime()))
```

4. Make sure pointing at singularity container created by Duke OIT to use LS-DYNA license. Only need to change folder name (TI_attempt3 here)

5. To make your own Field II Loads instead of defaults

```
field2dyna('nodes.dyn', 'field_params_L74_F0p52_focus20mm.json', 'elems.dyn')
```

6. Need two types of .json's now: field_params and probe definition (generally in gitlab>ultrasound>probes repo)

```
(base) AnnaKnight:FEM annaknight$ cat field_params_L74_F0p52_focus20mm.json
{
    "transducer": "l74",
    "transducer_type": "focused_multirow",
    "alpha_dB_cm_MHz": 0.8,
    "fnum": 0.52,
    "focus_m": [0.0, 0.0, 0.02],
    "center_focus_m": [0.0, 0.0, 0.0],
    "freq_MHz": 4.0,
    "impulse": "gaussian",
    "sound_speed_m_s": 1540,
    "sampling_freq_Hz": 100e6,
    "threads": 1,
    "lownslow": "true"
}
```

```
(base) AnnaKnight:json annaknight$ cat l74.json
{
    "transducerType": "focused_multirow",
        "noElements": 128.0,
        "noSubY": 25.1799,
        "noSubX": 1.0,
        "kerf": 0.02e-3,
        "height": 0.007,
        "elvFocus": 0.025,
        "width": 0.278e-3,
        "pitch": 2.9800e-4,
        "Rfocus": 0.025,
        "fractionalBandwidth": 70.0,
        "impulseResponse":
        {
                "f0": 5000000.0,
                "phase": 0.0,
                "bw": 70.0,
                "wavetype": "gaussian"
        },
        "noElementsY": 1.0,
        "centerFrequency": 5.0e6
}
```

This makes 'dyna-l-f4.00-F0.5-FD0.020-a0.80.mat'

FEM Tools Workflow

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

2. Launch a bash script

```
[aek27@dcc-login-01  /work/aek27/Testing
#!/bin/bash
#SBATCH -o /work/aek27/TestingFemTools2
#SBATCH -e/work/aek27/TestingFemTools20
#SBATCH --mem=64G
#SBATCH --partition=ultrasound
#SBATCH --exclude=dcc-ultrasound-01
#SBATCH --cpus-per-task=12


date
hostname

module load Python/3.8.1
source /work/aek27/.venv/bin/activate

python runTIsim.py
```

3. Standard Python script to make nodes, elems, bc, pmls, etc.

```
[aek27@dcc-login-01  /work/aek27/TestingFemTools2021/TI_attempt3 $ cat runTIsim.py
from os import environ, system
from socket import gethostname
from time import ctime

from fem.mesh import GenMesh, bc
from fem.mesh.GaussExc import generate_loads
from fem.post.create_disp_dat import create_dat as create_disp_dat
from fem.post import create_res_sim

print('STARTED: {}'.format(ctime()))
print('HOST: {}'.format(gethostname()))

##DYNADECK = 'gauss_qsym_pml.dyn'
DYNADECK = 'ortho_tilt11_rot0.dyn'
NTASKS = environ.get('SLURM_NTASKS', '8')

GenMesh.run((-2.0, 2.0, -2.0, 2.0, -4.0, 0.0), (160, 160, 160))

# setup no-symmetry condition
pml_elems = ((5, 5), (5, 5), (5, 5))
face_constraints = (('1,0,0,0,1,1', '1,0,0,0,1,1'),
                    ('0,1,0,1,0,1', '0,1,0,1,0,1'),
                    ('1,1,1,1,1,1', '1,1,1,1,1,1'))
edge_constraints = (((0, 1), (1, 0), (0, 0)), '1,1,0,1,1,1')
bc.apply_pml(pml_elems, face_constraints, edge_constraints)

generate_loads([0.25, 0.25, 0.75], [0.0, 0.0, -1.5])

##system('ls-dyna-d ncpu={} i={}'.format(NTASKS, DYNADECK))
system('singularity exec -p -B /work/aek27/TestingFemTools2021/TI_attempt3 /opt/apps/staging/ls-dyna-singularity/ls-dyna.sif ls-dyna-d ncpu={} i={} memory=600000000'.format(NTASKS, DYNADECK))



create_disp_dat()

create_res_sim.run(DYNADECK,
                   dispout="disp.dat",
                   ressim="res_sim.mat")
create_res_sim.run(dynadeck=DYNADECK,
                   dispout="disp.dat",
                   ressim='res_sim.h5')
create_res_sim.extract3Darfidata(dynadeck=DYNADECK,
                                 dispout="disp.dat",
                                 ressim="res_sim.pvd")

##os.system("xz -v disp.dat")

print('FINISHED: {}'.format(ctime()))
```

4. Make sure pointing at singularity container created by Duke OIT to use LS-DYNA license. Only need to change folder name (TI_attempt3 here)

5. To make your own Field II Loads instead of defaults

```
field2dyna('nodes.dyn', 'field_params_L74_F0p52_focus20mm.json', 'elems.dyn')
```

6. Need two types of .json's now: field_params and probe definition (generally in gitlab>ultrasound>probes repo)

```
(base) AnnaKnight:FEM annaknight$ cat field_params_L74_F0p52_focus20mm.json
{
    "transducer": "l74",
    "transducer_type": "focused_multirow",
    "alpha_dB_cm_MHz": 0.8,
    "fnum": 0.52,
    "focus_m": [0.0, 0.0, 0.02],
    "center_focus_m": [0.0, 0.0, 0.0],
    "freq_MHz": 4.0,
    "impulse": "gaussian",
    "sound_speed_m_s": 1540,
    "sampling_freq_Hz": 100e6,
    "threads": 1,
    "lownslow": "true"
}
```

```
(base) AnnaKnight:json annaknight$ cat l74.json
{
    "transducerType": "focused_multirow",
        "noElements": 128.0,
        "noSubY": 25.1799,
        "noSubX": 1.0,
        "kerf": 0.02e-3,
        "height": 0.007,
        "elvFocus": 0.025,
        "width": 0.278e-3,
        "pitch": 2.9800e-4,
        "Rfocus": 0.025,
        "fractionalBandwidth": 70.0,
        "impulseResponse":
        {
                "f0": 5000000.0,
                "phase": 0.0,
                "bw": 70.0,
                "wavetype": "gaussian"
        },
        "noElementsY": 1.0,
        "centerFrequency": 5.0e6
}
```

This makes 'dyna-l-f4.00-F0.5-FD0.020-a0.80.mat'

7. makeLoadsTemps('dyna*.mat', 'dyna*.mat', MeasuredISPPA, PulseDuration, SpecificHeat, Elementvolume, symmetry, loadcurveID)

Makes PointLoads-f4.00-F0.5-FD0.020-a0.80.dyn

FEM Tools Workflow

1. Make a venv (python virtual environment) and install fem

```
cd /work/yournetID
module load Python/3.8.1
python3.8 -m venv .venv
```

```
pip install git+https://github.com/mlp6/fem.git
```

2. Launch a bash script

```
[aek27@dcc-login-01  /work/aek27/Testing
#!/bin/bash
#SBATCH -o /work/aek27/TestingFemTools2
#SBATCH -e/work/aek27/TestingFemTools20
#SBATCH --mem=64G
#SBATCH --partition=ultrasound
#SBATCH --exclude=dcc-ultrasound-01
#SBATCH --cpus-per-task=12


date
hostname

module load Python/3.8.1
source /work/aek27/.venv/bin/activate

python runTIsim.py
```

3. Standard Python script to make nodes, elems, bc, pmls, etc.

```
[aek27@dcc-login-01  /work/aek27/TestingFemTools2021/TI_attempt3 $ cat runTIsim.py
from os import environ, system
from socket import gethostname
from time import ctime

from fem.mesh import GenMesh, bc
from fem.mesh.GaussExc import generate_loads
from fem.post.create_disp_dat import create_dat as create_disp_dat
from fem.post import create_res_sim

print('STARTED: {}'.format(ctime()))
print('HOST: {}'.format(gethostname()))

##DYNADECK = 'gauss_qsym_pml.dyn'
DYNADECK = 'ortho_tilt11_rot0.dyn'
NTASKS = environ.get('SLURM_NTASKS', '8')

GenMesh.run((-2.0, 2.0, -2.0, 2.0, -4.0, 0.0), (160, 160, 160))

# setup no-symmetry condition
pml_elems = ((5, 5), (5, 5), (5, 5))
face_constraints = (('1,0,0,0,1,1', '1,0,0,0,1,1'),
                    ('0,1,0,1,0,1', '0,1,0,1,0,1'),
                    ('1,1,1,1,1,1', '1,1,1,1,1,1'))
edge_constraints = (((0, 1), (1, 0), (0, 0)), '1,1,0,1,1,1')
bc.apply_pml(pml_elems, face_constraints, edge_constraints)

generate_loads([0.25, 0.25, 0.75], [0.0, 0.0, -1.5])

##system('ls-dyna-d ncpu={} i={}'.format(NTASKS, DYNADECK))
system('singularity exec -p -B /work/aek27/TestingFemTools2021/TI_attempt3 /opt/apps/staging/ls-dyna-singularity/ls-dyna.sif ls-dyna-d ncpu={} i={} memory=600000000'.format(NTASKS, DYNADECK))


create_disp_dat()

create_res_sim.run(DYNADECK,
                   dispout="disp.dat",
                   ressim="res_sim.mat")
create_res_sim.run(dynadeck=DYNADECK,
                   dispout="disp.dat",
                   ressim='res_sim.h5')
create_res_sim.extract3Darfidata(dynadeck=DYNADECK,
                                 dispout="disp.dat",
                                 ressim="res_sim.pvd")

##os.system("xz -v disp.dat")

print('FINISHED: {}'.format(ctime()))
```

4. Make sure pointing at singularity container created by Duke OIT to use LS-DYNA license. Only need to change folder name (TI_attempt3 here)

5. To make your own Field II Loads instead of defaults

```
field2dyna('nodes.dyn', 'field_params_L74_F0p52_focus20mm.json', 'elems.dyn')
```

6. Need two types of .json's now: field_params and probe definition (generally in gitlab>ultrasound>probes repo)

```
(base) AnnaKnight:FEM annaknight$ cat field_params_L74_F0p52_focus20mm.json
{
    "transducer": "l74",
    "transducer_type": "focused_multirow",
    "alpha_dB_cm_MHz": 0.8,
    "fnum": 0.52,
    "focus_m": [0.0, 0.0, 0.02],
    "center_focus_m": [0.0, 0.0, 0.0],
    "freq_MHz": 4.0,
    "impulse": "gaussian",
    "sound_speed_m_s": 1540,
    "sampling_freq_Hz": 100e6,
    "threads": 1,
    "lownslow": "true"
}
```

```
(base) AnnaKnight:json annaknight$ cat l74.json
{
    "transducerType": "focused_multirow",
        "noElements": 128.0,
        "noSubY": 25.1799,
        "noSubX": 1.0,
        "kerf": 0.02e-3,
        "height": 0.007,
        "elvFocus": 0.025,
        "width": 0.278e-3,
        "pitch": 2.9800e-4,
        "Rfocus": 0.025,
        "fractionalBandwidth": 70.0,
        "impulseResponse":
        {
                "f0": 5000000.0,
                "phase": 0.0,
                "bw": 70.0,
                "wavetype": "gaussian"
        },
        "noElementsY": 1.0,
        "centerFrequency": 5.0e6
}
```

This makes 'dyna-l-f4.00-F0.5-FD0.020-a0.80.mat'

7. makeLoadsTemps('dyna*.mat', 'dyna*.mat', MeasuredISPPA, PulseDuration, SpecificHeat, Elementvolume, symmetry, loadcurveID)

Makes PointLoads-f4.00-F0.5-FD0.020-a0.80.dyn

8. Insert in place of loads.dyn in main dyna deck (ortho_tilt11_rot0.dyn')

```
*INCLUDE
./nodes.dyn
*INCLUDE
./elems.dyn
*INCLUDE
./mat_002_ortho_tilt11_rot0_aopt2.dyn
*PART

1,1,1,0,0,0,0
*SECTION_SOLID
1,1
*INCLUDE
./bc.dyn
*INCLUDE
./PointLoads-f4.00-F0.5-FD0.020-a0.80.dyn
```