BB Binary workflow

SUMMARY

- Old HF workflow
  - losing 10% core hours for formatting text
  - losing 250% storage space for storing txt
- New BB workflow
  - checks LF/HF compatibility, previous case of different `nt` no longer undetected
  - kupe 80 processes = 55 seconds, hypocentre 28 processes = 15 seconds (siteamp not fully vectorised)
  - quickly finished, not fully tested yet
  - No longer losing 600% storage space for storing both Vel and Acc as text

How to run

HF

```
mpirun -n 80 python2 hf_sim.py source.stoch stations.ll HF.bin \ # HF.bin is the output file
-m /home/nesi00213/VelocityModel/Mod-1D/Cant1D_v1-midQ.1d \ # override vm
--dt 0.02
-i # individually run stations
```

BB

```
mpirun -n 80 python2 bb_sim.py LF/bevan2012_v3_s103252/OutBin /path/to/vm/ HF.bin BB.bin # BB.bin is the output
file
```

BB gets hf_vs_ref from HF file (stored as part of HF), lf_vs_ref from vs3d.file.s

Python interface

HF (and BB)

```
from qcore.timeseries import HFSeis
# load file
hf = HFSeis('HF.bin')
# retrieve timeseries (x,y,z)
hf.acc('CACS')
# only x
hf.acc('CACS', comp = <0, or hf.X or hf.COMP['090']>)
# store station as txt
hf.acc2txt('CACS')
# store all stations as txt, same as if ran standard txt version
hf.all2txt(prefix = 'Acc/hf_')
```

LF

```
# just like for vel
lf.vel('CACS')
# you can access transparently converted accelerations
lf.acc('CACS')
```