Test params for SIRE using LHC FT nominal params

Parameters	Nominal (BCMS) for 10 MV (2016)
E [GeV]	6500
$\varepsilon_{x,y}$ [μm]	2.5
4σ bunch length [ns]	1.0
Bunch population [10^{11}]	1.1

Aug&July17

Oct17

Check what happens if instead of having the full SIRE distribution we cut at 3sigma and at 2sigma

full

```
ans =

   0.252000000000000                   0   0.001927201452045   0.001600000000000

ans =

   0.259000000000000   0.001000000000000   0.012951534210891   0.014600000000000

ans =

   0.251000000000000                   0   0.998000000000000   0.003000000000000   0.001600000000000

ans =

   0.243000000000000                   0   0.833000000000000   0.004000000000000   0.002000000000000
```

3sigma

```
ans =

   0.252000000000000                   0   0.001927159269239   0.001800000000000

ans =

   0.259000000000000   0.001000000000000   0.012970993683451   0.015500000000000

ans =

   0.251000000000000                   0   0.998000000000000   0.003000000000000   0.001700000000000

ans =

   0.244000000000000                   0   0.843000000000000   0.004000000000000   0.002400000000000
```

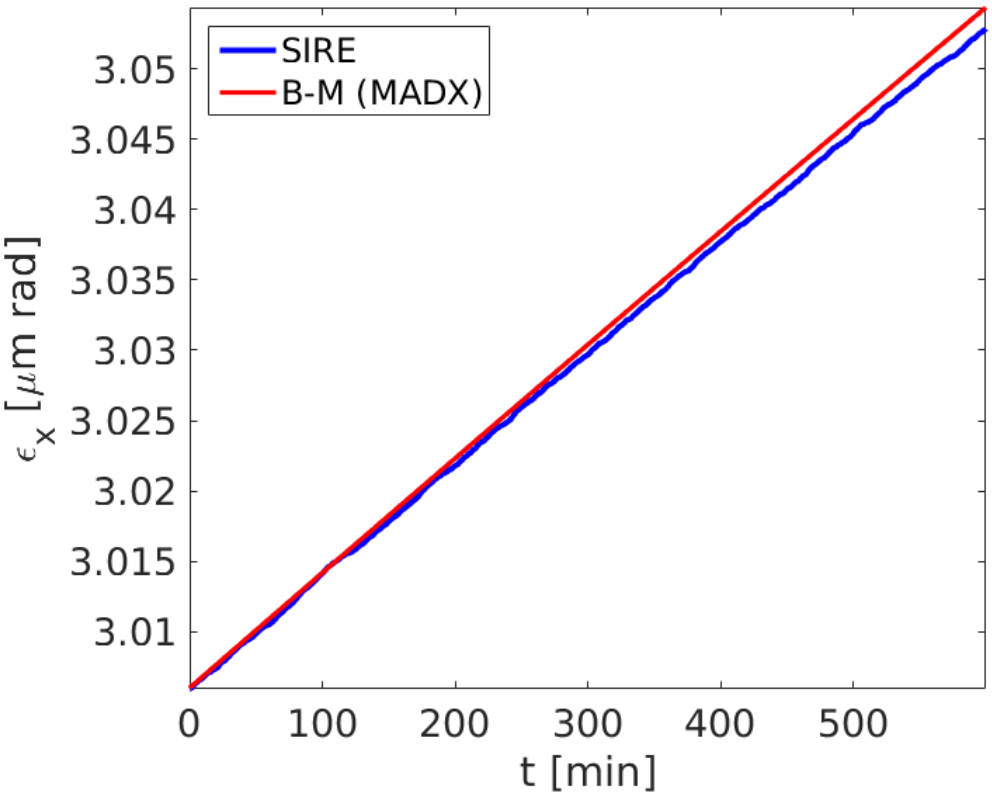
2sigma

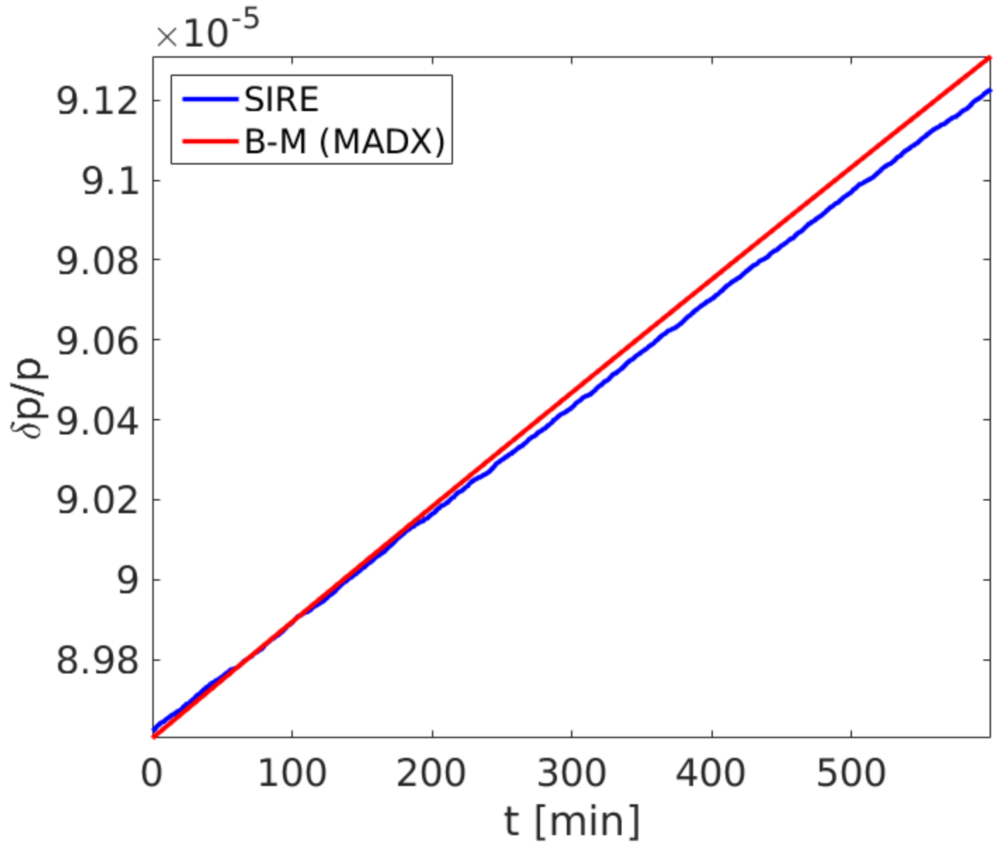
```
ans =

   0.252000000000000   1.359000000000000   0.001891441069689   0.349100000000000

ans =

   0.262000000000000   0.266000000000000   0.013159564470582   0.289700000000000

ans =

   0.251000000000000                   0   0.996000000000000   0.004000000000000   0.002000000000000

ans =

   0.242000000000000   0.001000000000000   0.818000000000000   0.007000000000000   0.002200000000000
```

Nominal (BCMS)

test_lessIBS: only IBS, #mp=200000, #mp/cell=5

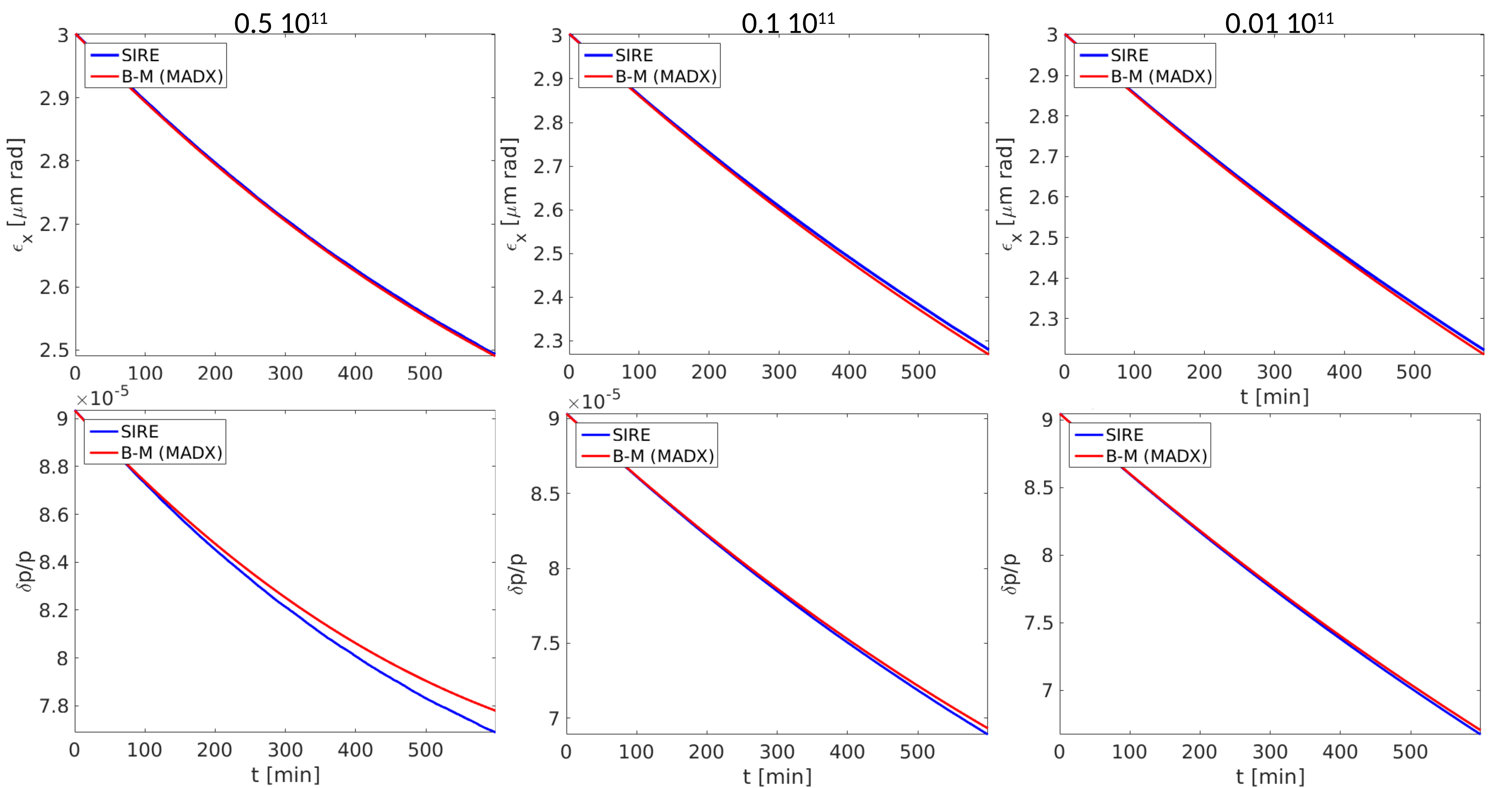
Parameters	Nominal (BCMS) for 10 MV (2016)
E [GeV]	6500
$\varepsilon_{x,y}$ [μm]	3.0
4σ bunch length [ns]	1.0
Bunch population [10^{11}]	0.1





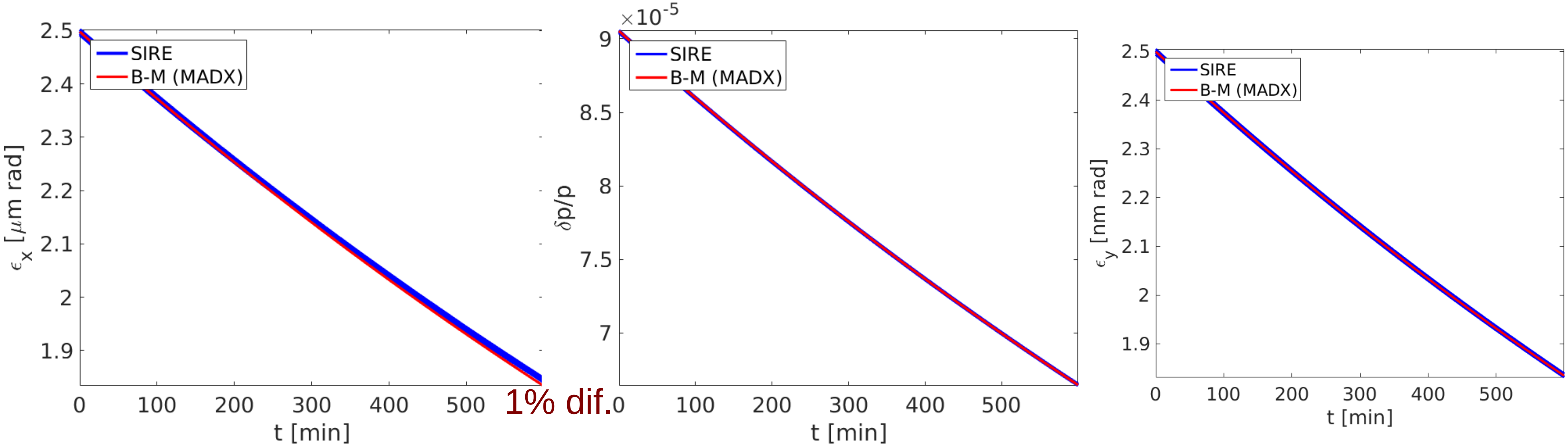
test_lessIBS: IBS&SR, #mp=175000, #mp/cell=5

Parameters @ FB	Nominal (BCMS) for 10 MV (2016)
E [GeV]	6500
$\varepsilon_{x,y}$ [μm]	3.0
4σ bunch length [ns]	1.0
Bunch population [10^{11}]	0.5 , 0.1 and 0.01



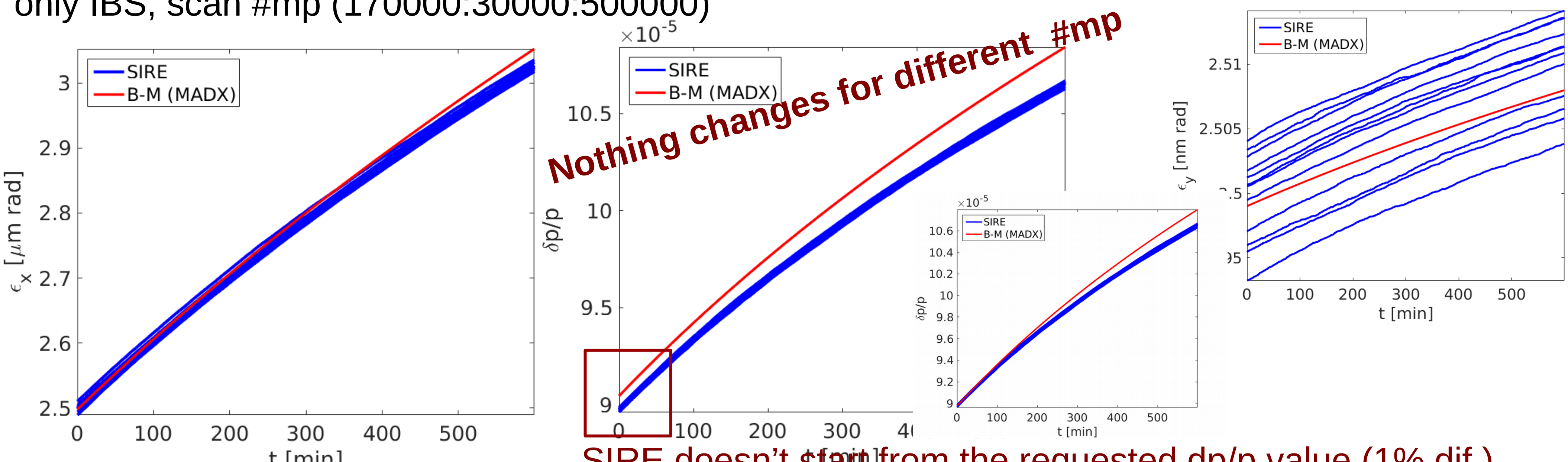
Aug17

only SR



1% dif.

only IBS, scan #mp (170000:30000:500000)



Nothing changes for different #mp

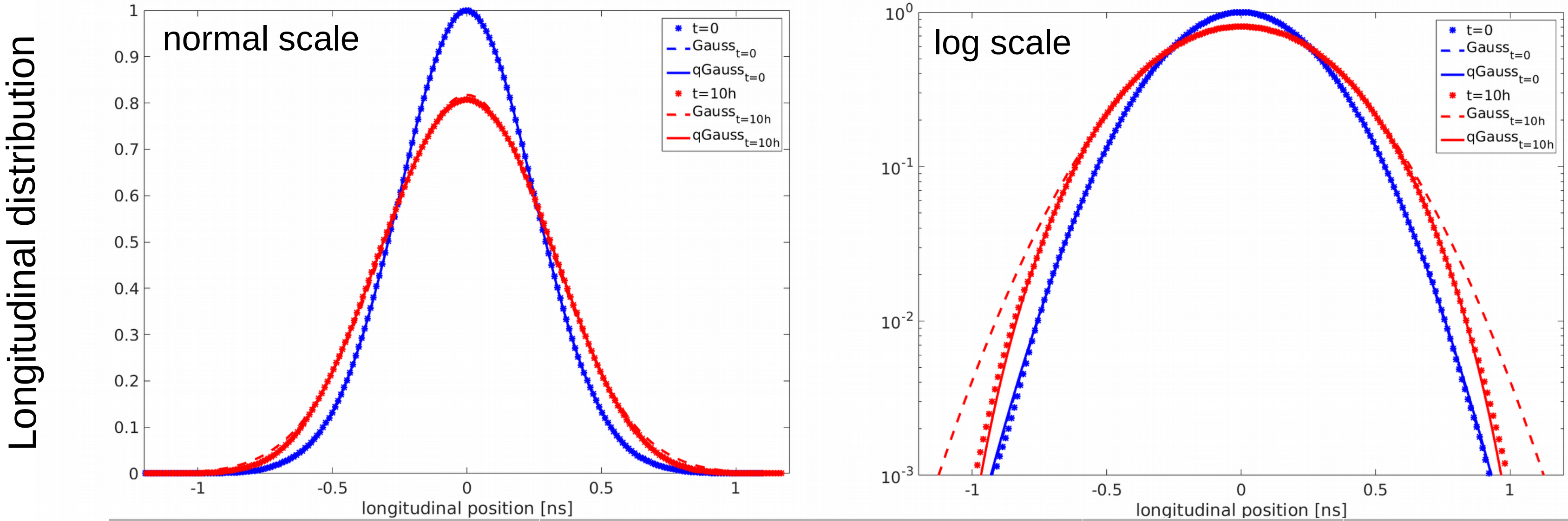
SIRE doesn't start from the requested dp/p value (1% dif.) and that is not because of the random generator!So why?

BECAUSE I need to calculate it based on el=2(dp/p)^2 (no damping) . If I use that everything is ok

MADX assumes always a Gaussian, however SIRE updates the distribution→ that can explain the dif. Observed (see next slide).

only IBS

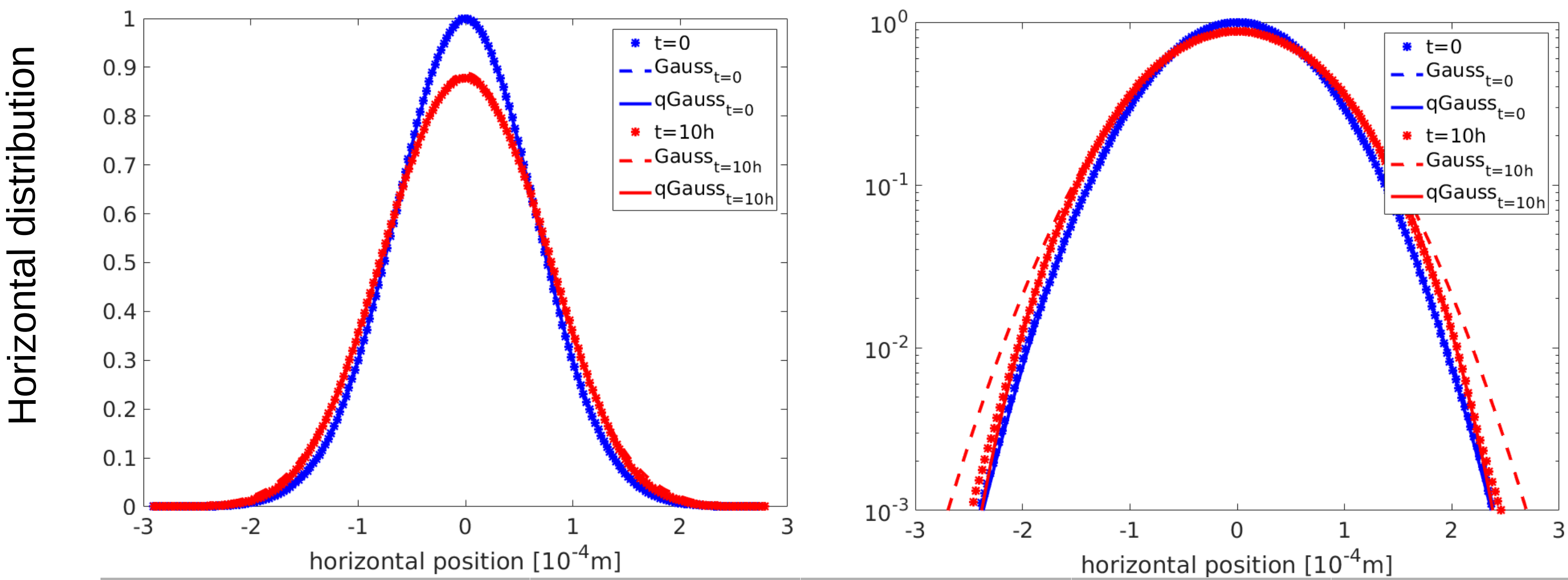
Bunch profile evolution in all planes as computed by SIRE



	q initial	sigma initial	q final	sigma final
SIRE qGaussian	1.010±0.003	0.250±0.001	0.874±0.003	0.295±0.001

rmse=0.001

rmse=0.002



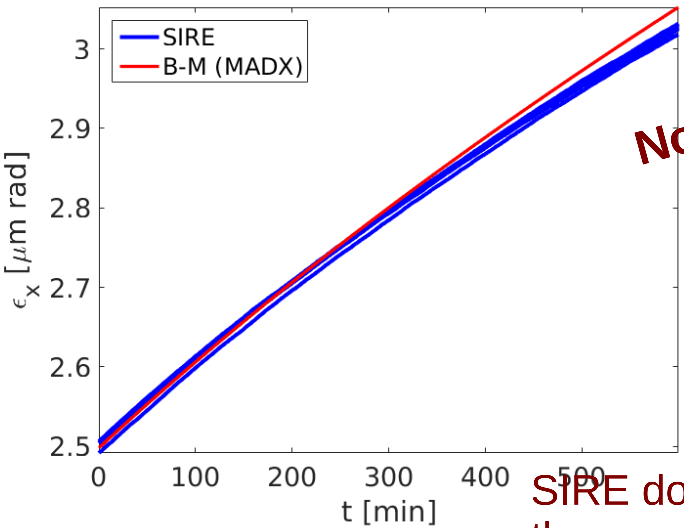
	q initial	sigma initial	q final	sigma final
SIRE qGaussian	0.995±0.003	0.643±0.002	0.872±0.004	0.702±0.003

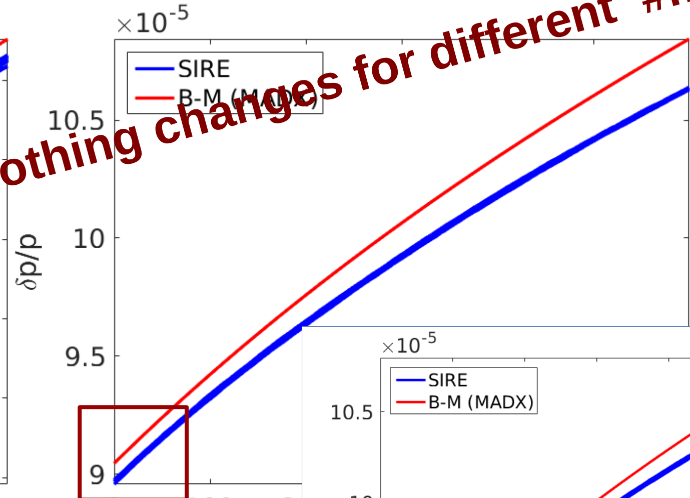
rmse=0.001

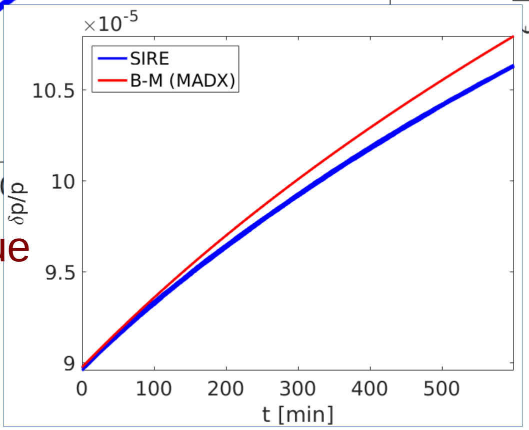
rmse=0.001

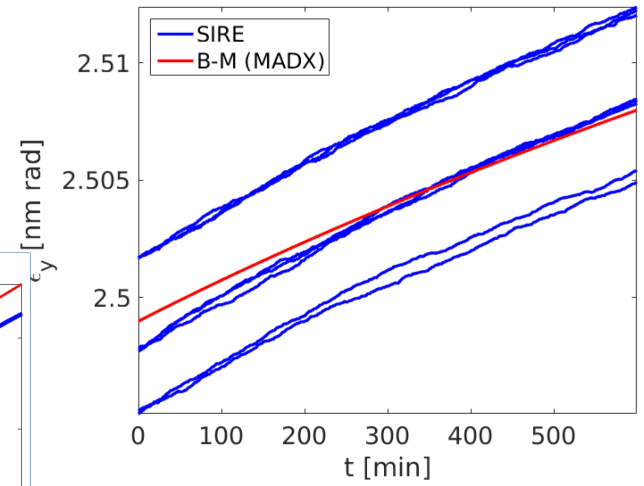
only IBS, scan #mp/cell (4:2:26)

Nothing changes for different #mp/cell









SIRE doesn't start from the requested dp/p value (1% dif.) and that is not because of the random generator!So why?

BECAUSE I need to calculate it based on el=2(dp/p)^2 (no damping) . If I use that everything is ok!

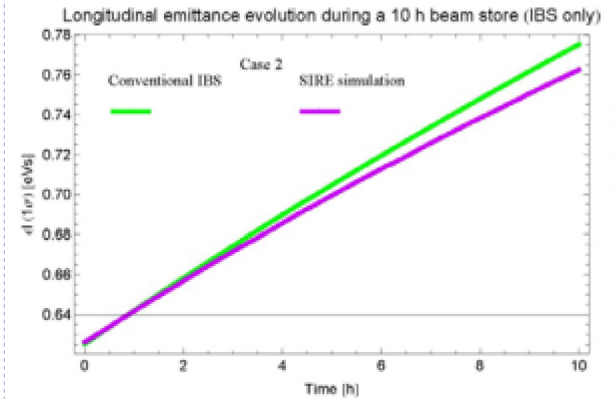


Fig. 5: Comparison of ε_L between SIRE and straight IBS computations (case 2, Table 1): difference $\delta_{max}(\Delta\varepsilon_L/\varepsilon_L) \sim 2\%$.

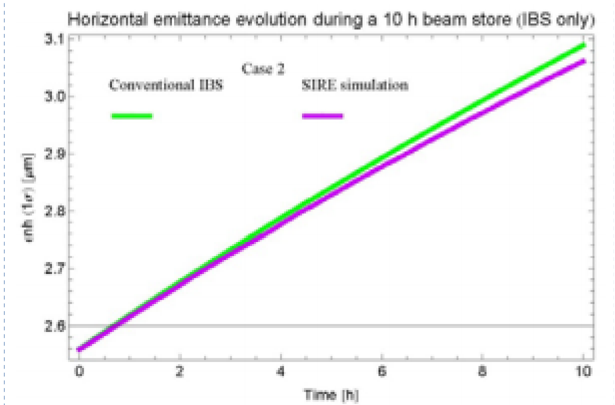
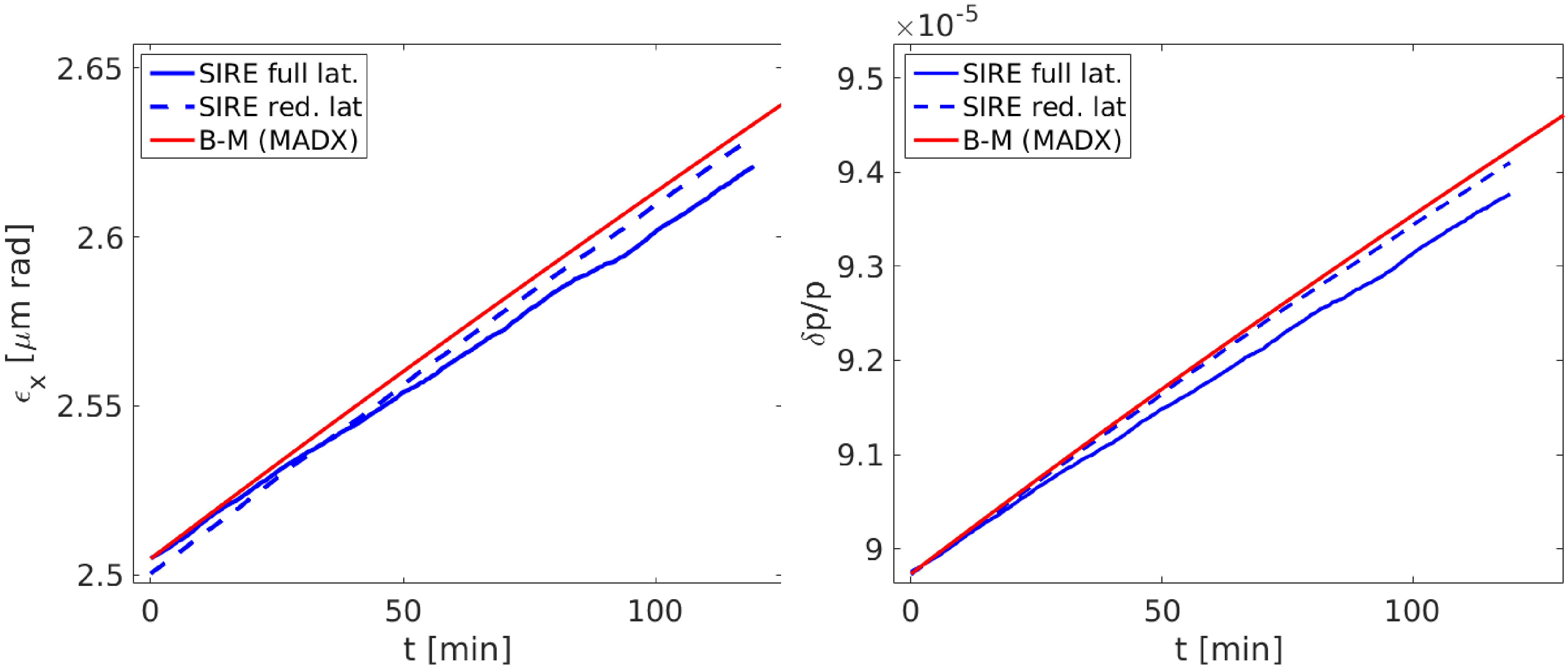


Fig. 6: Comparison of $\varepsilon_{N,H}$ between SIRE and straight IBS computations (case 2, Table 1): difference $\delta_{max}(\Delta\varepsilon_H/\varepsilon_H) \sim 1\%$.

test_wholelattice_2h: only IBS, #mp=50000, #mp/cell=5, for 2h



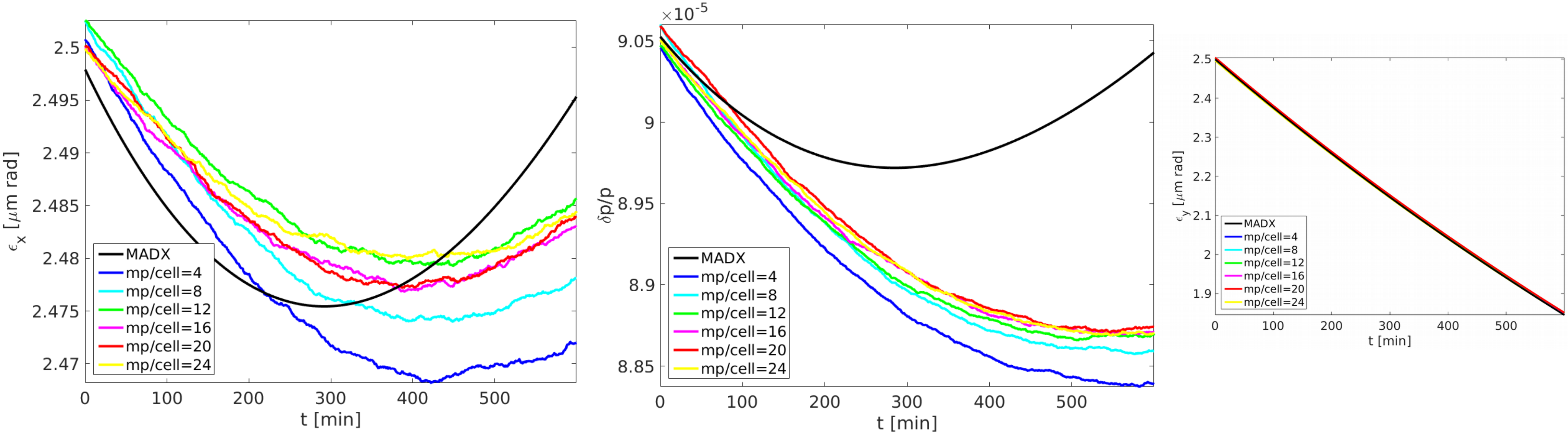
The SIRE reduced lattice is closer to the full lattice MADX result, than the SIRE full lattice is, why? The issue is why the full and the reduced lattice results disagree? At FB, there was not such a disagreement.

compare **reduced to full lattice** for 1h at FB and at FT, nominal params

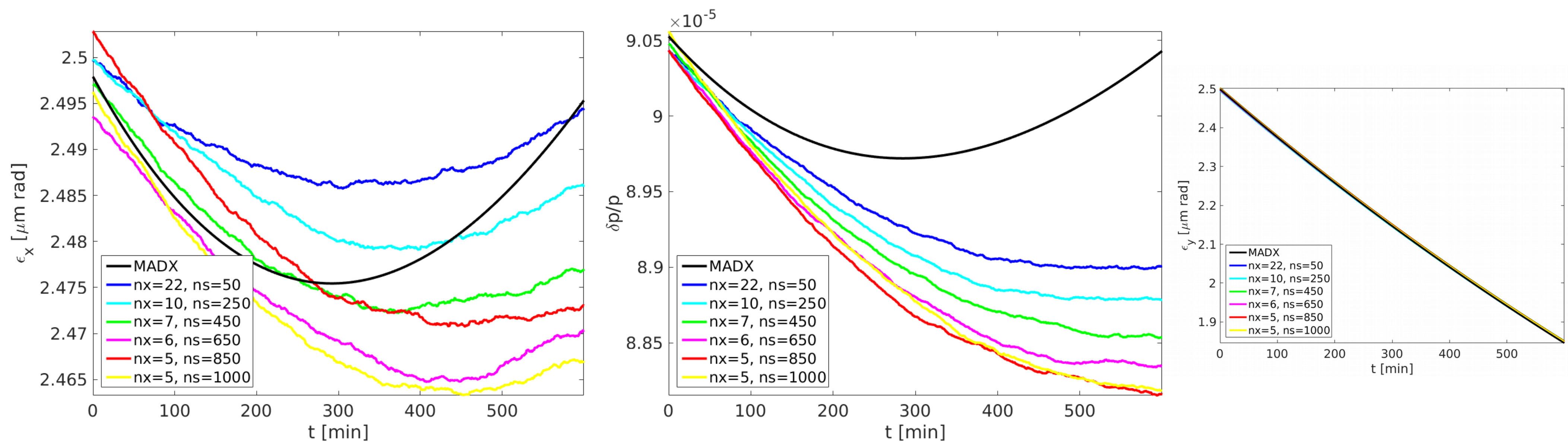
1h at FB

1h at FT

IBS+SR+QE, #mp=500000, scan #mp/cell (4:2:26)

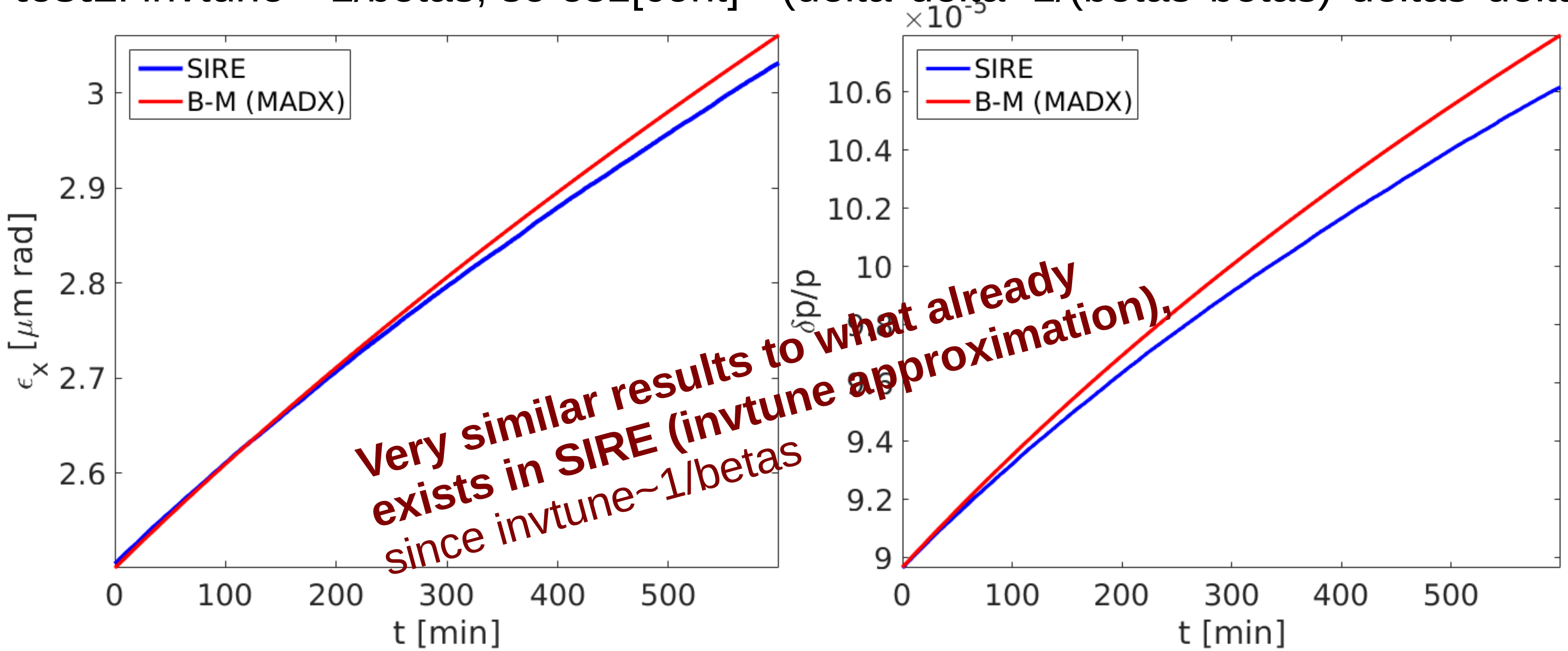


IBS+SR+QE, #mp=500000, #mp/cell=20, scan #ncells (50:50:1000)



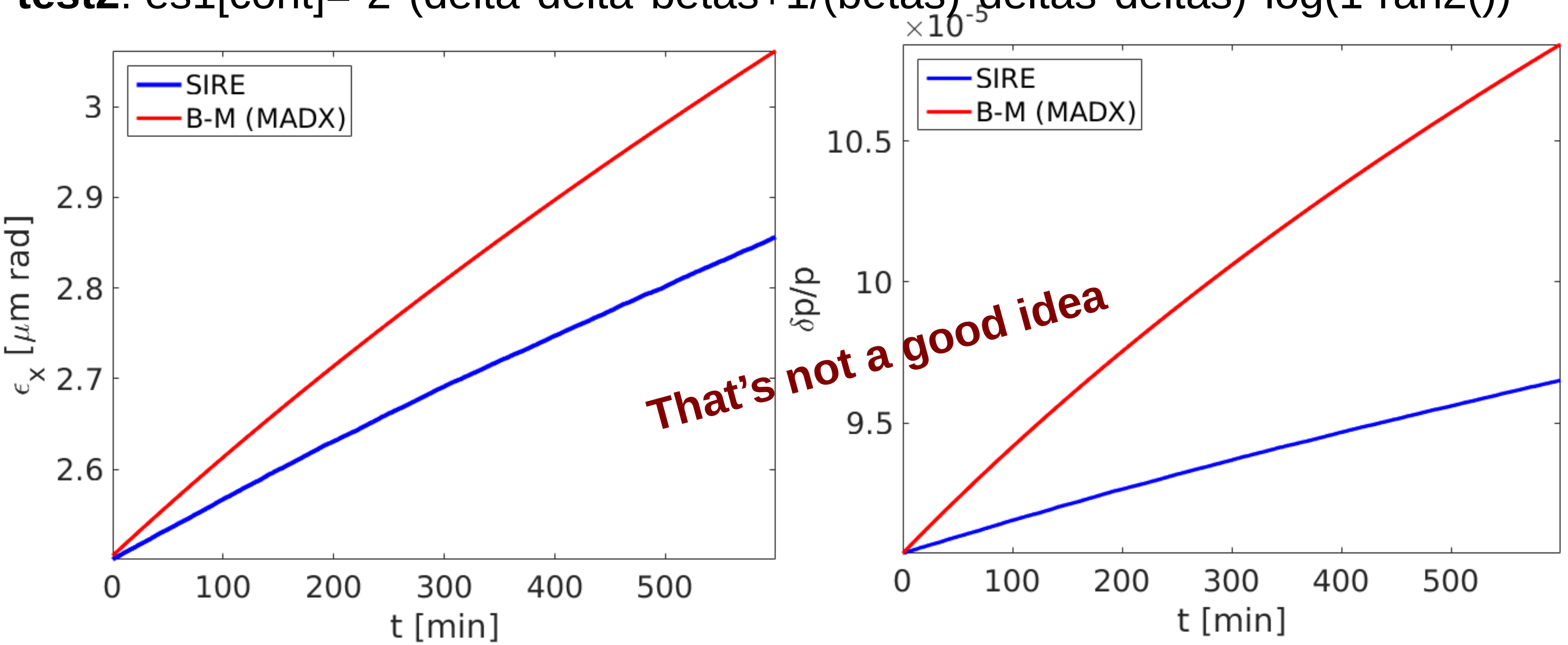
test longitudinal plane in SIRE

test1: invtune→1/betas, so es1[cont]=-(delta*delta+1/(betas*betas)*deltas*deltas)*log(1-ran2())



Very similar results to what already exists in SIRE (invtune approximation), since invtune~1/betas

test2: es1[cont]=-2*(delta*delta*betas+1/(betas)*deltas*deltas)*log(1-ran2())

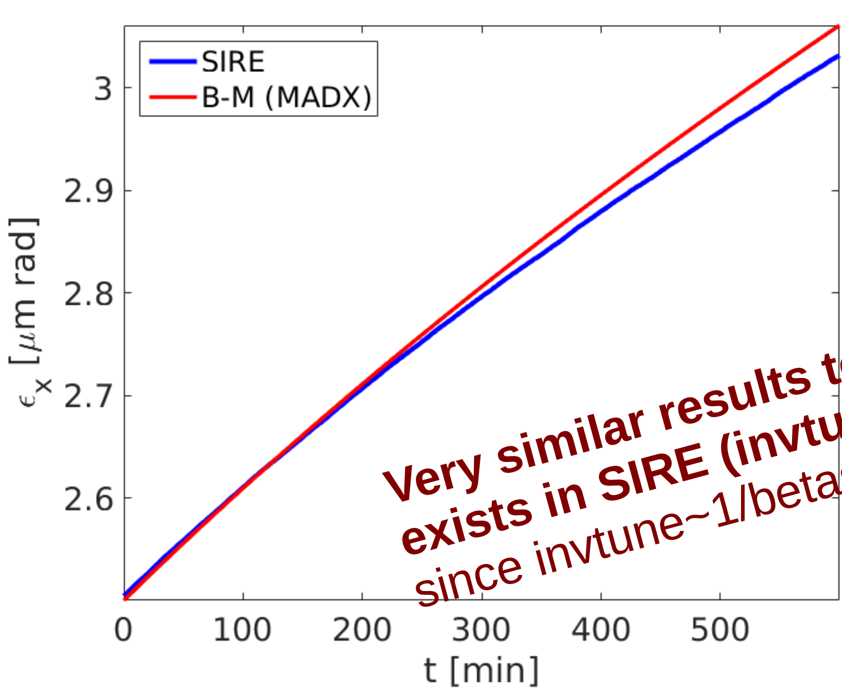


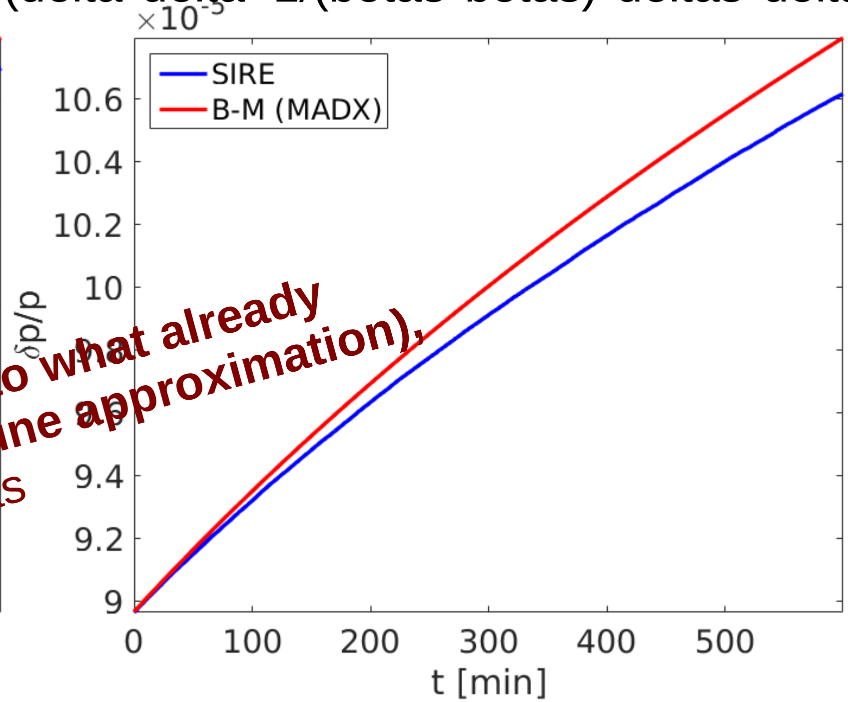
That's not a good idea

from Martini 2016 paper (pg 18) → check my notes

test longitudinal plane in SIRE

test1: invtune→1/betas, so es1[cont]=-(delta*delta+1/(betas*betas)*deltas*deltas)*log(1-ran2())

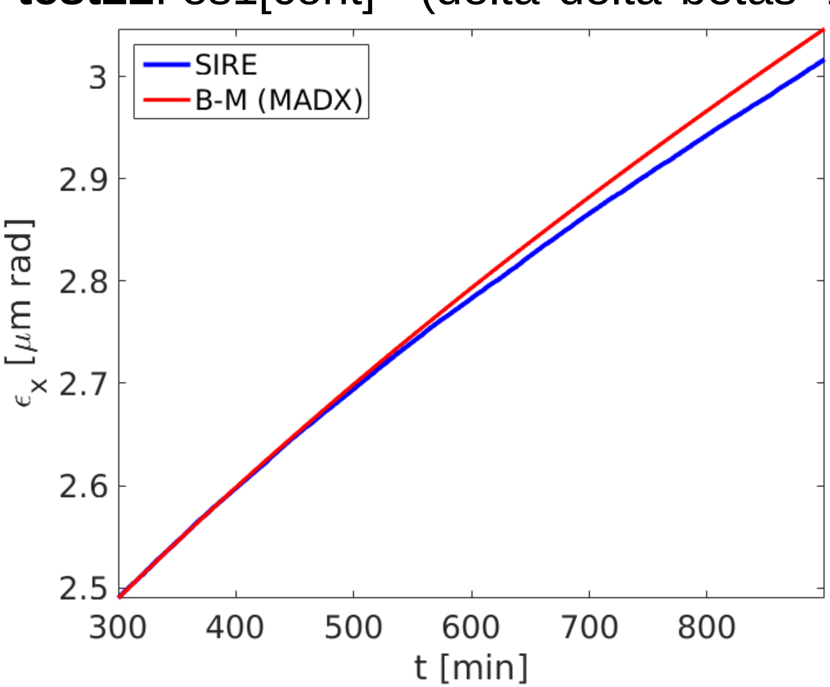


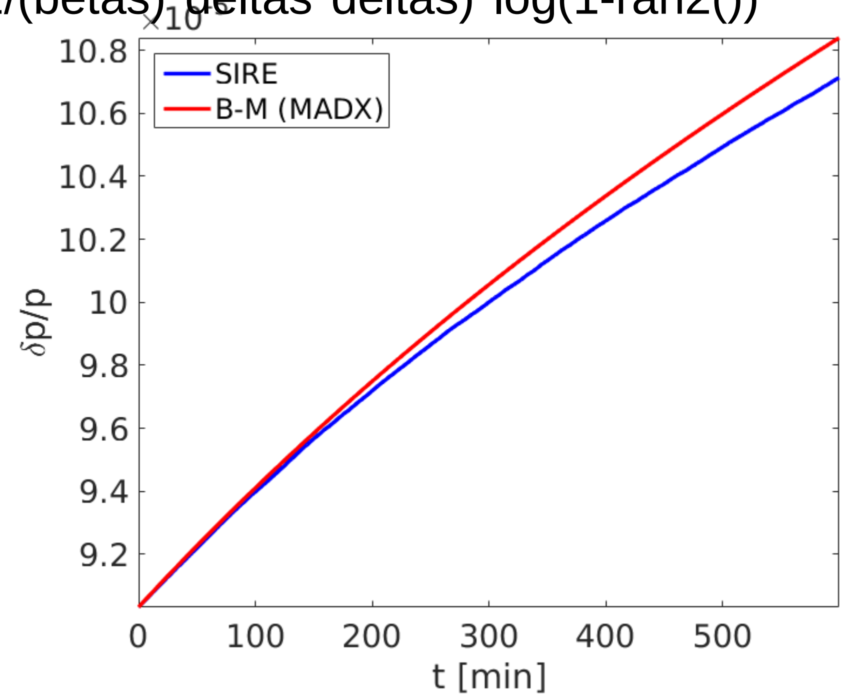


1.7% dif.

Very similar results to what already exists in SIRE (invtune approximation),
since invtune~1/betas

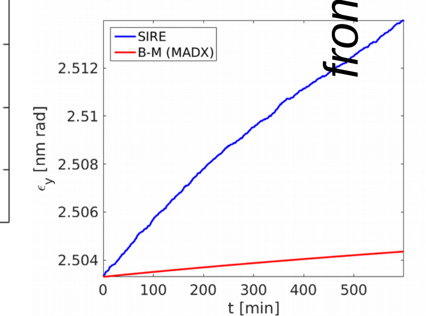
test22: es1[cont]=-(delta*delta*betas+1/(betas)*deltas*deltas)*log(1-ran2())





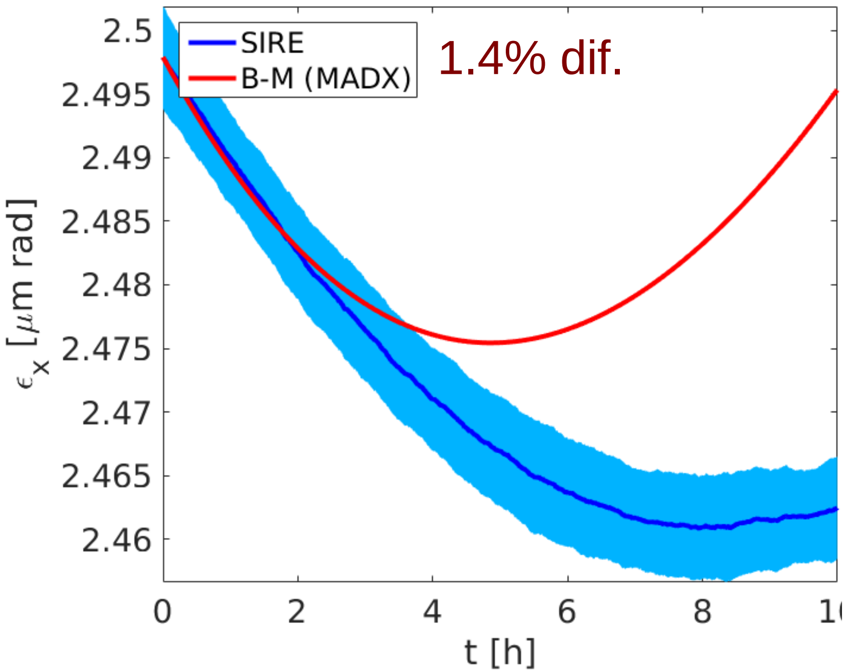
1.2% dif.

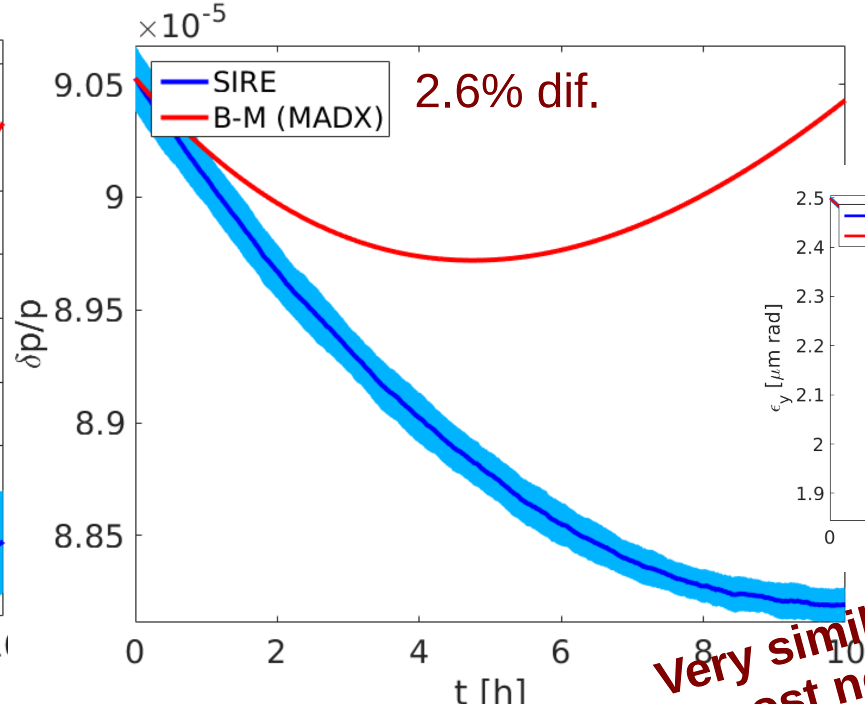
from Martini 2016 paper (pg 18) → check my notes

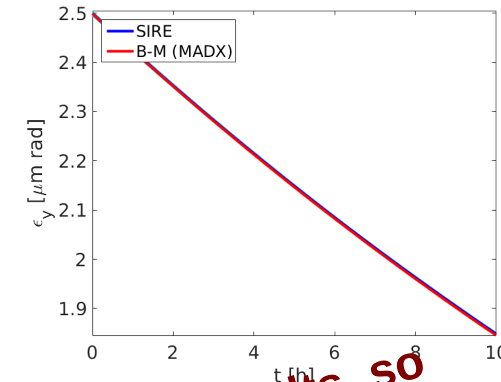


IBS+SR+QE test longitudinal plane in SIRE

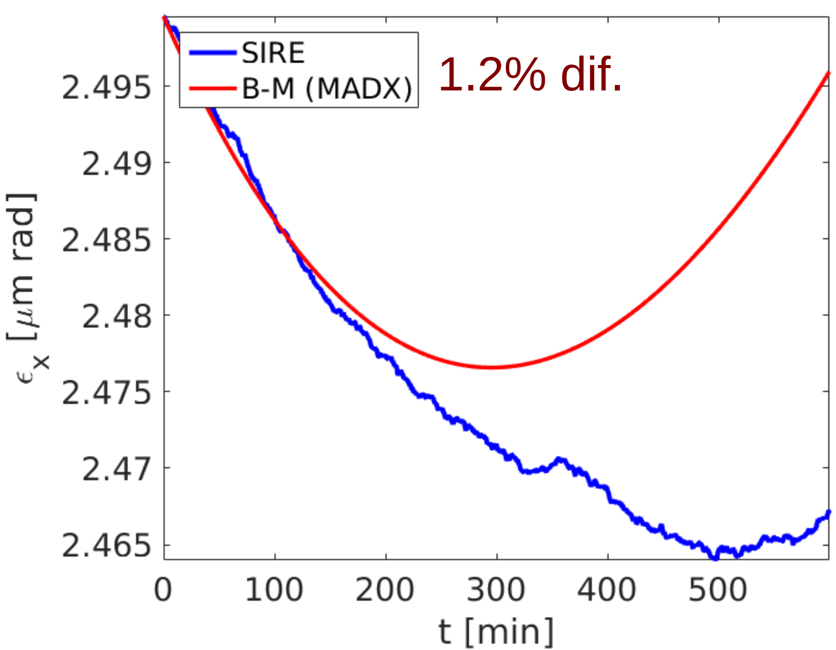
current SIRE: invtune approximation

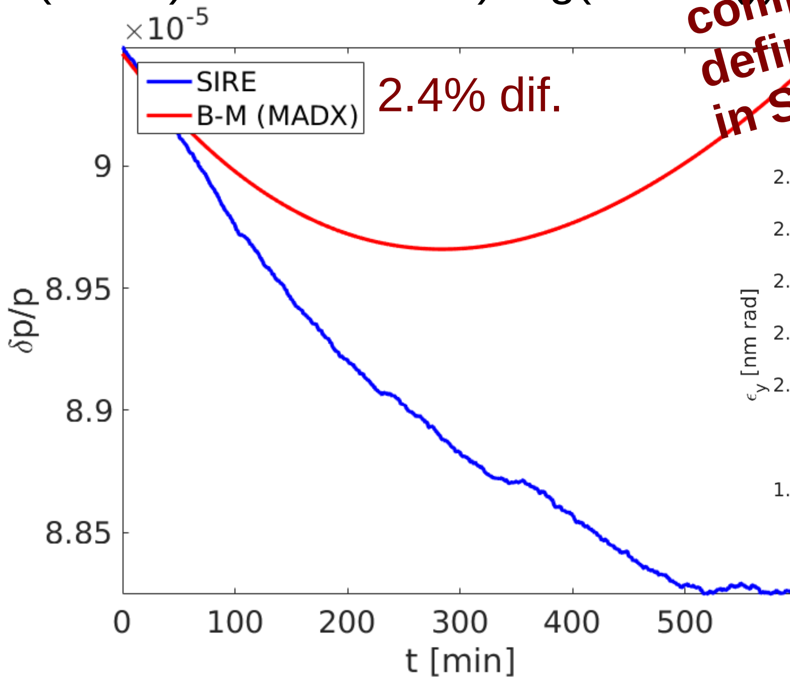


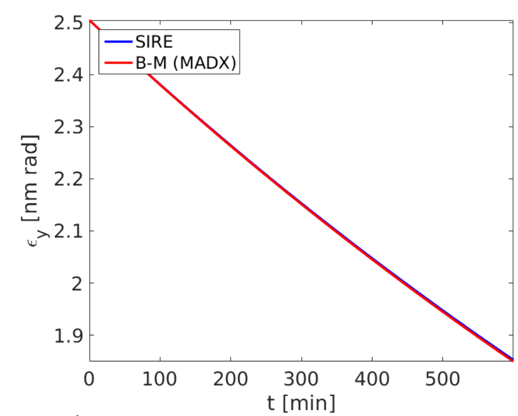




test22: es1[cont]=-(delta*delta*betas+1/(betas)*deltas*deltas)*log(1-ran2())







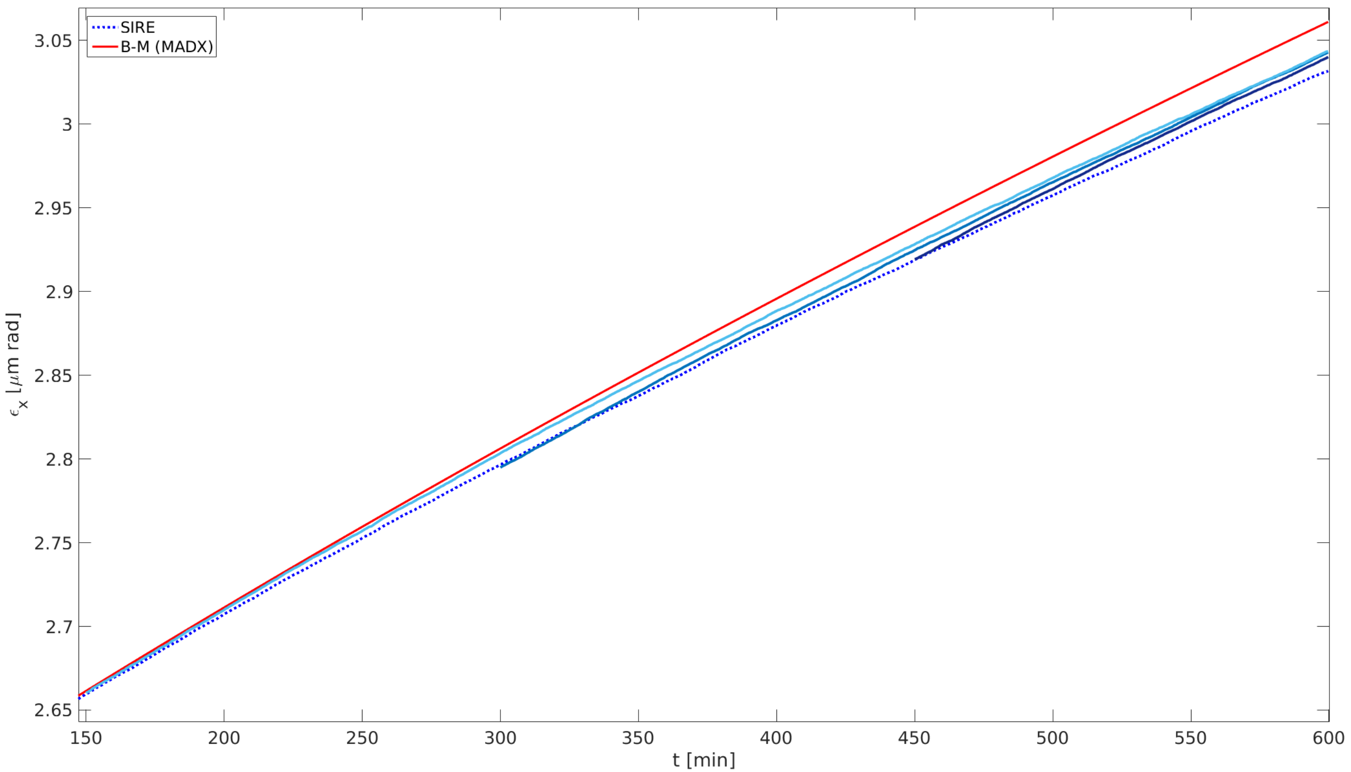
Very similar results, so almost no improvement compared to the existing definition of longit. plane in SIRE, and so???

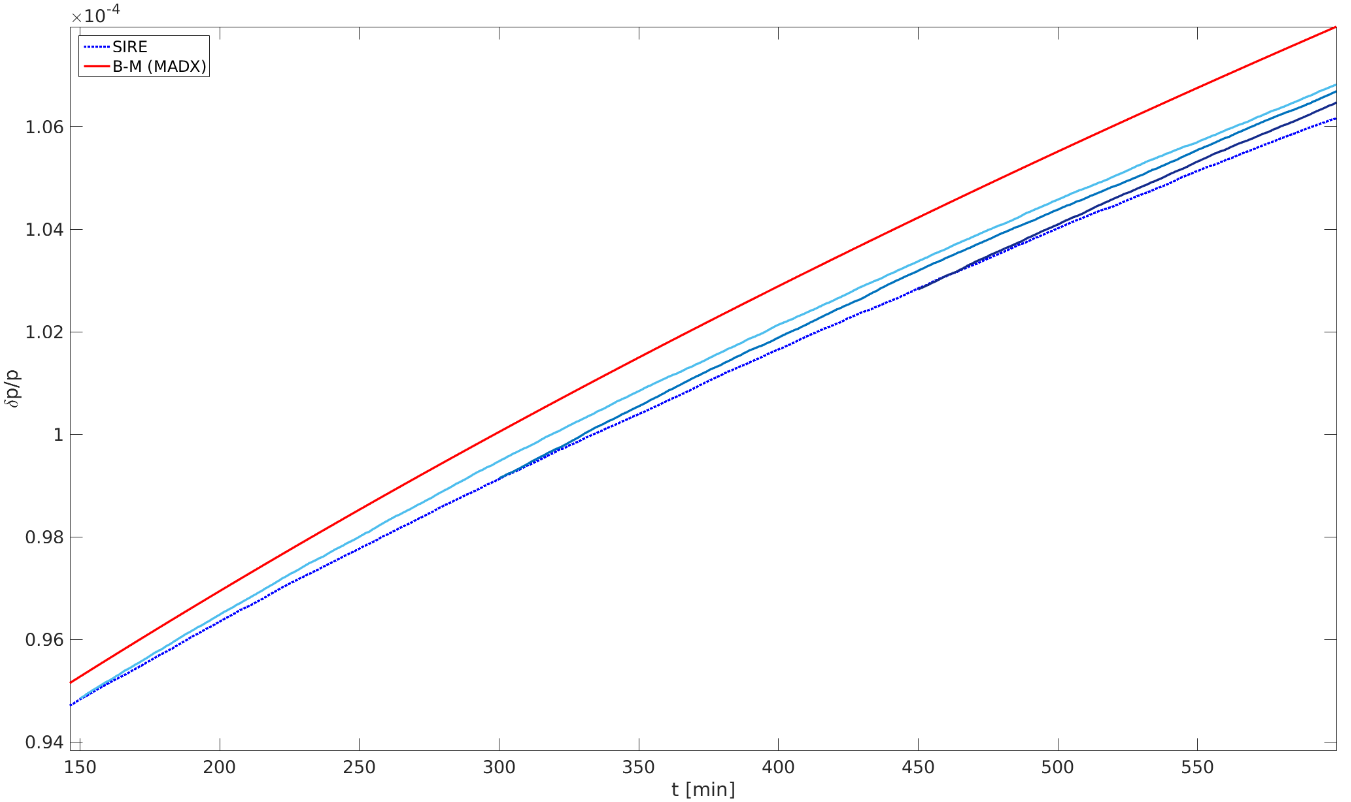
test longitudinal plane in SIRE

test1: invtune→1/betas, so es1[cont]=-(delta*delta+1/(betas*betas)*deltas*deltas)*log(1-ran2()) . This test gives very similar results to what already exists in SIRE (invtune approximation), since invtune~1/betas.
Here: track in SIRE initially Gaussian distributions that correspond to the emit&enspread(and bunch length) values taken during specific timepoints within 10 h full tracking time in SIRE and the MADX results, that are not the same. Basically, I apply Gaussian distributions 2.5, 5 and 7.5 hours after the beginning, having in all planes the beam sizes that correspond to the values I see during the full 10h SIRE tracking and to the MADX results at this timepoints. So, in total I run 6 different cases for the timepoints mentioned (2.5, 5 and 7.5 hours), 3 based on the SIRE values and another 3 based on the MADX values. In this way we can see whether the discrepancy between sire and madx comes from the distribution form or not.

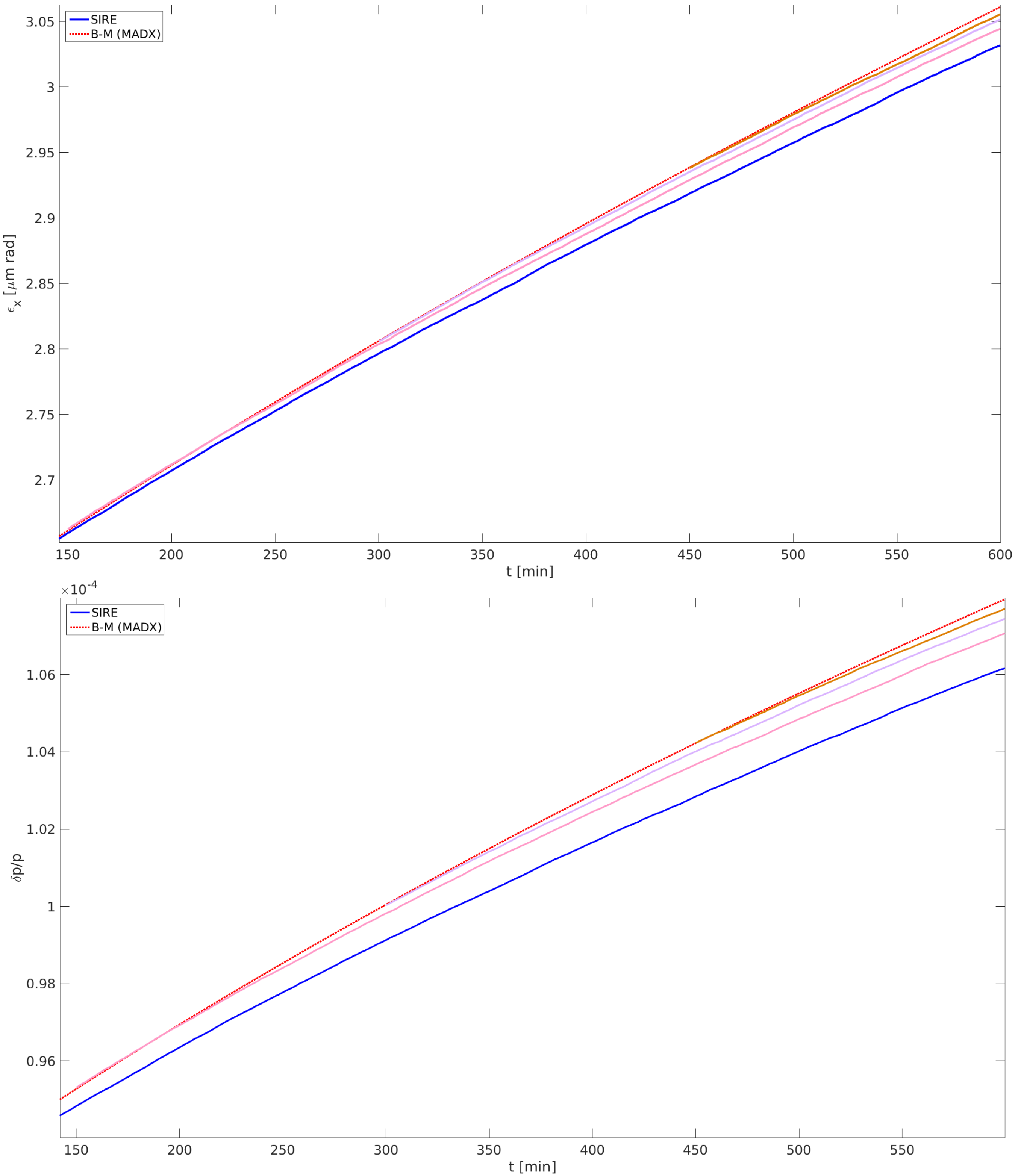
See the 2 following slides

track in SIRE initially Gaussian distributions that correspond to the emit&enspread (and bunch length) values taken during specific timepoints within 10 h full tracking time **in SIRE**



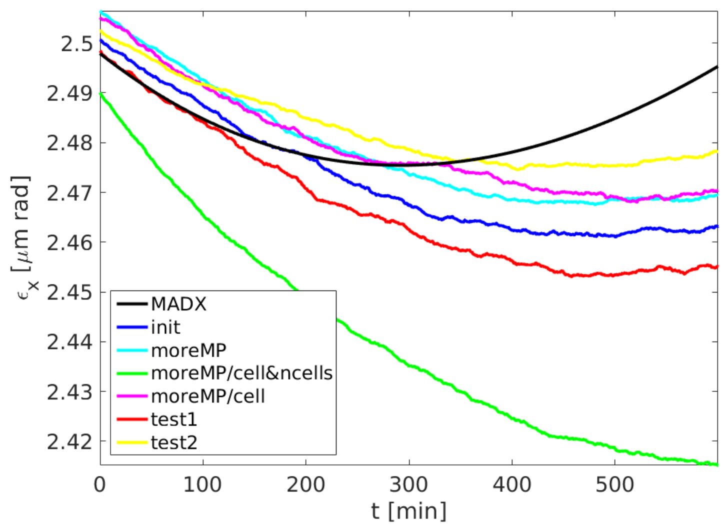


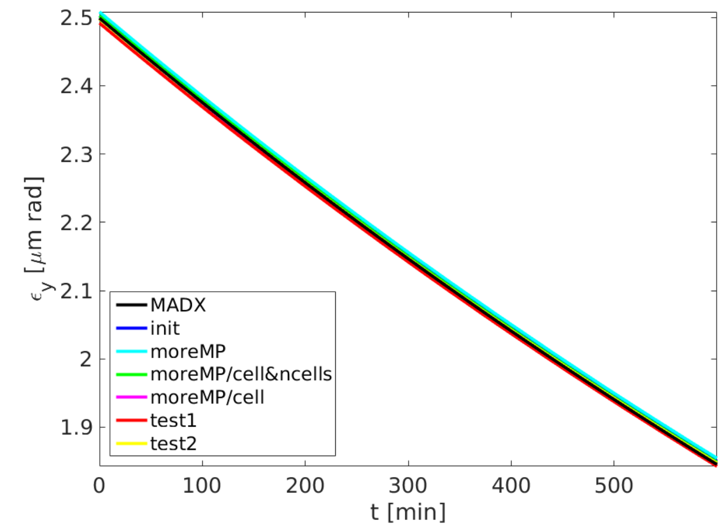
track in SIRE initially Gaussian distributions that correspond to the emit&enspread(and bunch length) values taken during specific timepoints within 10 h **from MADX** results

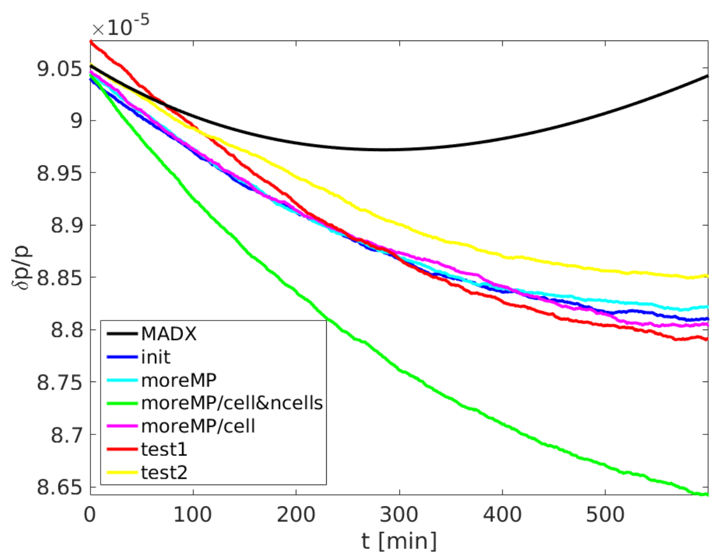


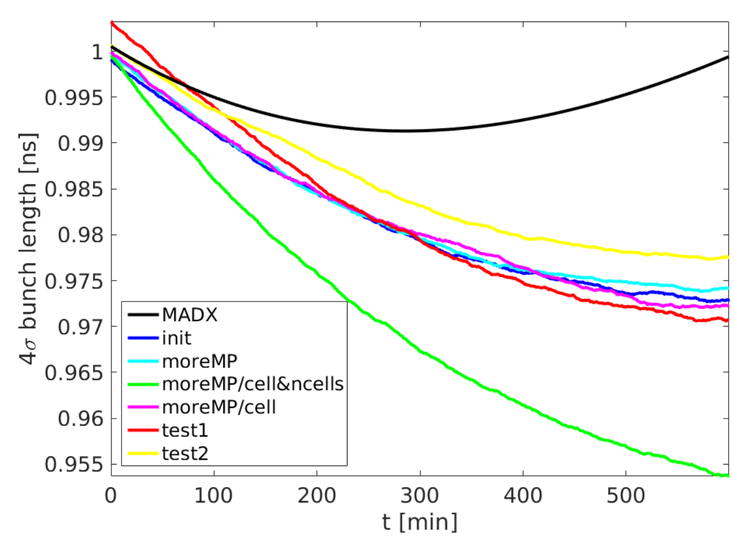
July17

files/parameters	#mp	#mp/cell	ncellx*	ncells
Paramstest-forscanCells	175000	5	10	350
params_moreMP	250000	5	12	350
params_moreMPpercell	175000	15	6	350
params_moreMPpercell_ncells**	175000	87	2	500
test1	175000	20	4	500
test2	250000	10	10	250







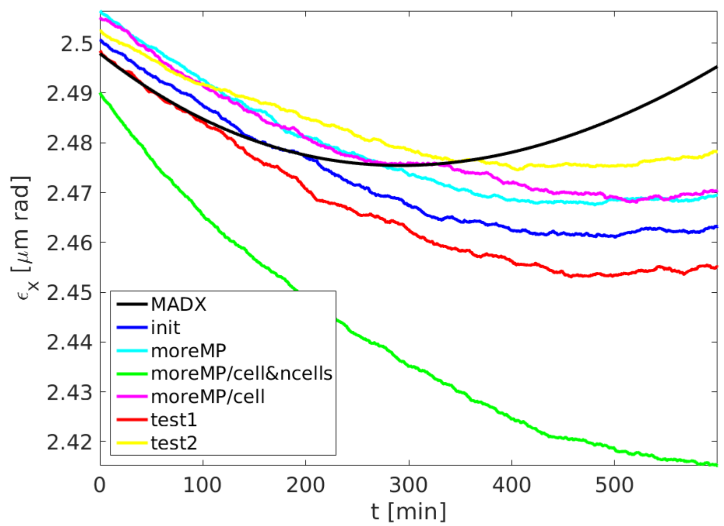


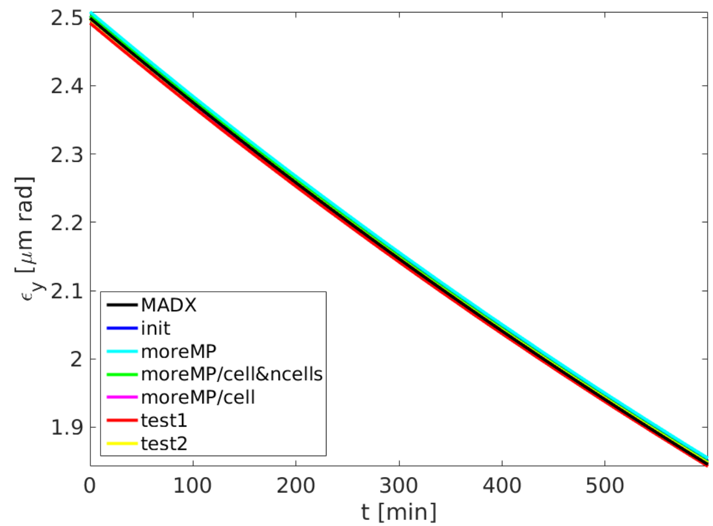
files/parameters	#mp	#mp/cell	ncellx*	ncells
Paramstest-forscanCells	175000	5	10	350
params_moreMP	250000	5	12	350
params_moreMPpercell	175000	15	6	350
params_moreMPpercell_ncells**	175000	87	2	500
test1	175000	20	4	500
test2	250000	10	10	250

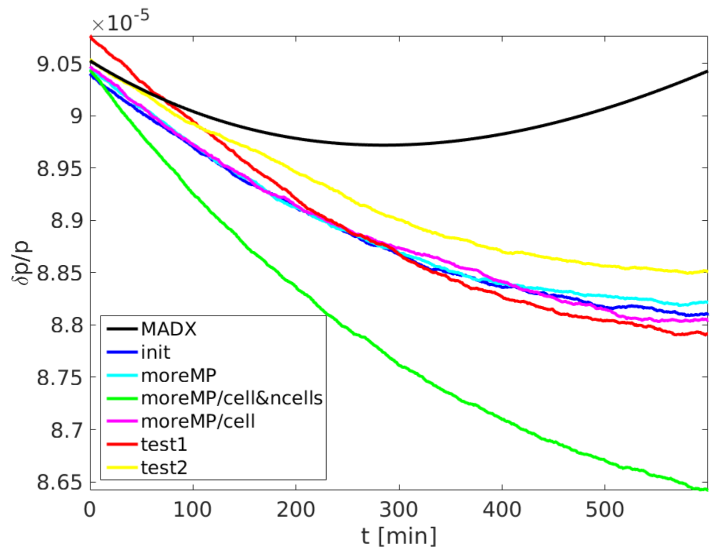
-test3 and tes6 see the #cells defined in the params file, without taking into account the #mp/cell
-test7 is for ncellz=ceil((**numpart***ratio/(**ncells***ncellx***mppercell**)));
-All the other tests are for ncellx=sqrt((**numpart***ratio/(**ncells***mppercell**)));
ncellz=ceil(**ncellx**/ratio);

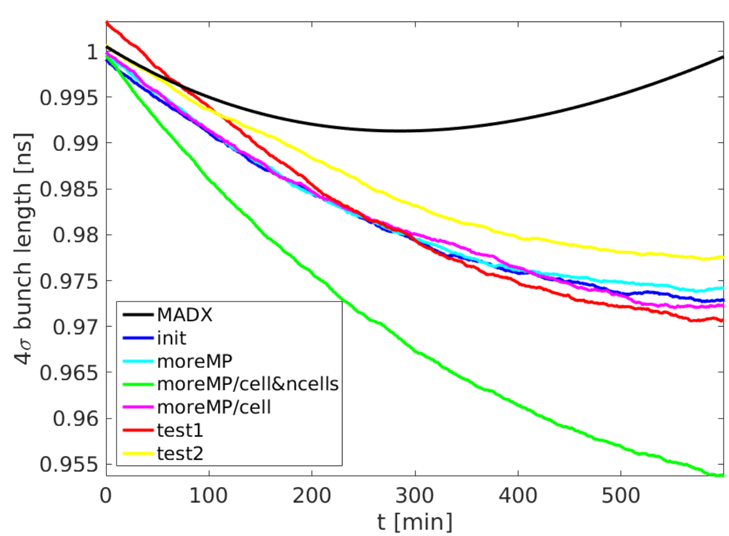
*based on the ncellx=sqrt(#mp*ratio/(ncells*#mp/cell)), where the ratio is the sigmax/sigmay.
**the bl/sigmax~250 then for ncells=500 it should be ncellx=87, so, for #mp=175000, #mp/cell=2

files/parameters	#mp	#mp/cell	ncellx*	ncells
Paramstest-forscanCells	175000	5	10	350
params_moreMP	250000	5	12	350
params_moreMPpercell	175000	15	6	350
params_moreMPpercell_ncells**	175000	87	2	500
test1	175000	20	4	500
test2	250000	10	10	250

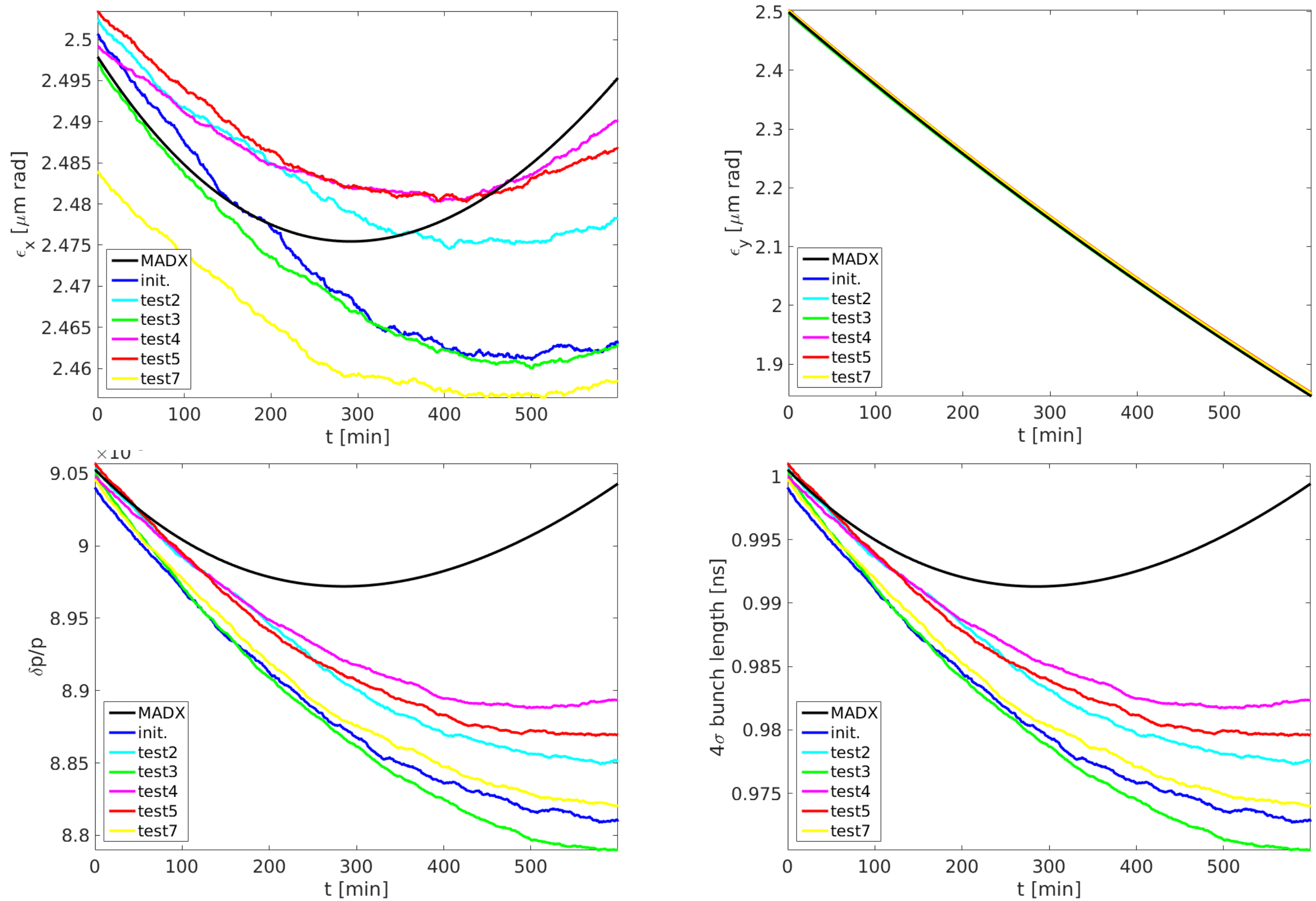








files/parameters	#mp	#mp/cell	ncellx*	ncells
Paramstest-forscanCells	175000	5	10	350
test2	250000	10	10	250
test3	400000	-	20, 20	1000
test4	500000	20	16	100
test5	350000	15	10	250
test6 (no)	300000	-	10, 10	500
test7	300000	10	10	500



```
ex1[cont]=-2*epsx*log(1-ran2());     // -2*exmean*log(1-ran2()) --> the factor of 2 is to go from emittance to action
ez1[cont]=-2*epsz*log(1-ran2());
es1[cont]=-(delta*delta+invtune*invtune*deltas*deltas)*log(1-ran2());          //          this is alsready in action
phix1[cont]=2*pi*ran2();
phiz1[cont]=2*pi*ran2();
phis1[cont]=2*pi*ran2();
```

```
ratio=sqrt((betax[0]*exm[KINJ]+dispx[0]*dispx[0]*delta*delta)/(betaz[0]*ezm[KINJ]+dispz[0]*dispz[0]*delta*delta));
        ncellx=ceil(sqrt((numpart*ratio/(ncells*mppercell))));
        ncellz=ceil(ncellx/ratio);
```

```
phix[comodo]=2.0*pi*ran2();
phiz[comodo]=2.0*pi*ran2();
phis[comodo]=2.0*pi*ran2();
//  double *x,*xp,*z,*zp,*deltasp,*deltap
deltap[comodo]=sqrt(es[comodo])*cos(phis[comodo])+qes1*eqdelta*sqrt(1-exp(-DTIME/dtimes));
deltasp[comodo]=sqrt(es[comodo])*sin(phis[comodo])/invtune+qes2*eqdeltas*sqrt(1-exp(-DTIME/dtimes));
x[comodo]=sqrt(ex[comodo]*betax[i])*cos(phix[comodo])+qex1*sqrt(betax[i]*eqx*(1-exp(-DTIME/dtimex)));
xp[comodo]=-sqrt(ex[comodo]/betax[i])*(alphax[i]*cos(phix[comodo])+sin(phix[comodo]))+qex2*sqrt(eqx*(1+alphax[i]*alphax[i])/betax[i]*(1-exp(-DTIME/dtimex)));
z[comodo]=sqrt(ez[comodo]*betaz[i])*cos(phiz[comodo])+qez1*sqrt(betaz[i]*eqz*(1-exp(-DTIME/dtimez)));
zp[comodo]=-sqrt(ez[comodo]/betaz[i])*(alphaz[i]*cos(phiz[comodo])+sin(phiz[comodo]))+qez2*sqrt(eqz*(1+alphaz[i]*alphaz[i])/betaz[i]*(1-exp(-DTIME/dtimez)));;

ex[comodo]=betax[i]*xp[comodo]*xp[comodo]+2.0*alphax[i]*xp[comodo]*x[comodo]+(1+alphax[i]*alphax[i])/betax[i]*x[comodo]*x[comodo];
ez[comodo]=betaz[i]*zp[comodo]*zp[comodo]+2.0*alphaz[i]*zp[comodo]*z[comodo]+(1+alphaz[i]*alphaz[i])/betaz[i]*z[comodo]*z[comodo];
es[comodo]=deltap[comodo]*deltap[comodo]+invtune*invtune*deltasp[comodo]*deltasp[comodo];
phis[comodo]=atan(invtune*deltasp[comodo]/deltap[comodo]);
```

```
phix[cont]=2.0*pi*ran2();
phiz[cont]=2.0*pi*ran2();
phis[cont]=2.0*pi*ran2();
//  double *x,*xp,*z,*zp,*deltasp,*deltap
deltap[cont]=sqrt(es[cont])*cos(phis[cont]);
deltasp[cont]=sqrt(es[cont])*sin(phis[cont])/invtune;
x[cont]=sqrt(ex[cont]*betax[i])*cos(phix[cont])+dispx[i]*deltap[cont];
xp[cont]=-sqrt(ex[cont]/betax[i])*(alphax[i]*cos(phix[cont])+sin(phix[cont]))+disp1x[i]*deltap[cont];
z[cont]=sqrt(ez[cont]*betaz[i])*cos(phiz[cont])+dispz[i]*deltap[cont];
zp[cont]=-sqrt(ez[cont]/betaz[i])*(alphaz[i]*cos(phiz[cont])+sin(phiz[cont]))+disp1z[i]*deltap[cont];
```

```
    ex[cont]=betax[i]*(xp[cont]-disp1x[i]*deltap[cont])*(xp[cont]-disp1x[i]*deltap[cont])+2.0*alphax[i]*(xp[cont]-disp1x[i]*deltap[cont])*(x[cont]-dispx[i]*deltap[cont])
+(1+alphax[i]*alphax[i])/betax[i]*(x[cont]-dispx[i]*deltap[cont])*(x[cont]-dispx[i]*deltap[cont]);
    ez[cont]=betaz[i]*(zp[cont]-disp1z[i]*deltap[cont])*(zp[cont]-disp1z[i]*deltap[cont])+2.0*alphaz[i]*(zp[cont]-disp1z[i]*deltap[cont])*(z[cont]-dispz[i]*deltap[cont])
+(1+alphaz[i]*alphaz[i])/betaz[i]*(z[cont]-dispz[i]*deltap[cont])*(z[cont]-dispz[i]*deltap[cont]);
    es[cont]=deltap[cont]*deltap[cont]+invtune*invtune*deltasp[cont]*deltasp[cont];
    phis[cont]=atan(invtune*deltasp[cont]/deltap[cont]);
    if (deltap[cont]<0)
        {
          phis[cont]=phis[cont]+pi;
        }
    phix[cont]=atan(-(xp[cont]-disp1x[i]*deltap[cont])/(x[cont]-dispx[i]*deltap[cont])*betax[i]-alphax[i]);
    if (x[cont]-dispx[i]*deltap[cont]<0)
        {
          phix[cont]=phix[cont]+pi;
        }
    phiz[cont]=atan(-(zp[cont]-disp1z[i]*deltap[cont])/(z[cont]-dispz[i]*deltap[cont])*betaz[i]-alphaz[i]);
    if (z[cont]-dispz[i]*deltap[cont]<0)
        {
          phiz[cont]=phiz[cont]+pi;
```