```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE simulation [
<!ENTITY Npts     "64">
<!ENTITY Nsamples    "64">
<!ENTITY L  "3e-5">
]>
<simulation xmds-version="2">
  <name>GPE_1D_commandline_course</name>

  <author> Sebastian Wuester </author>
  <description>
    Gross-Pitaevskii-equation in 1D, SI units
  </description>

  <geometry>
    <propagation_dimension> t </propagation_dimension>
    <transverse_dimensions>
      <dimension name="x" lattice="&Npts;"  domain="(-&L;, &L;)" />
    </transverse_dimensions>
  </geometry>


  <features>
    <benchmark />
    <auto_vectorise />
    <fftw />
    <globals>
      <![CDATA[
      const double hbar = 1.05457266e-34;
      double omega = 10.0*(2.0*M_PI);           // to be modified by command line parameters
      const double omega_perp = 200.0*(2.0*M_PI);
      const double x0 = 0.0e-5;

      // Rb 87
      const double mass = 1.4432e-25;
      const double as = 5.5e-9;

      //dervied quantities
      const double Natoms = 100.0;

      const double U = 4.0*M_PI*hbar*hbar*as/mass;
      const double sigma_perp = sqrt(hbar/mass/omega_perp);
      const double U1d = U/(2.0*M_PI*sigma_perp*sigma_perp);

      double normfact = 0.0;    // get set after command line parameters are known
      double sigma = 0.0;
      ]]>
    </globals>
    <arguments>
       <argument name="trap_scale"  type="real" default_value="1.0"/>
       <argument name="width_scale" type="real" default_value="1.0"/>
       <![CDATA[
          omega *= sqrt(trap_scale);
          sigma = width_scale*sqrt(hbar/mass/omega);
          normfact = pow(M_PI*sigma*sigma,-0.25);
        ]]>
       </arguments>
  </features>

  <vector name="wavefunction" initial_space="x" type="complex">
    <components>psi</components>
```

```
    <initialisation>
      <![CDATA[
         const double delx = x - x0;
         psi = normfact*sqrt(Natoms)*exp(-0.5*delx*delx/sigma/sigma);
      ]]>
    </initialisation>
  </vector>

  <vector name="potentials" initial_space="x" type="real">
    <components>trap</components>
    <initialisation>
      <![CDATA[
        trap=0.5*mass*omega*omega*x*x;
      ]]>
    </initialisation>
  </vector>

  <computed_vector name="moments" dimensions="" type="real">
    <components> norm expecxx expecx  </components>
      <evaluation>
      <dependencies  basis="x"> wavefunction </dependencies>
      <![CDATA[
          norm = mod2(psi);
          expecxx = x*x*mod2(psi);
          expecx = x*mod2(psi);
      ]]>
    </evaluation>
  </computed_vector>

  <sequence>
    <integrate algorithm="ARK89" interval="0.4" tolerance="1e-8">
      <samples>200 200 200</samples>
      <operators>
        <operator kind="ip">
          <operator_names>L</operator_names>
          <![CDATA[
           L = -i*0.5*hbar*kx*kx/mass;
          ]]>
        </operator>
        <integration_vectors>wavefunction</integration_vectors>
        <dependencies>potentials</dependencies>
        <![CDATA[
         double dens=psi.Re()*psi.Re() + psi.Im()*psi.Im();

         dpsi_dt =  L[psi] - i*(U1d*dens + trap )*psi/hbar;
         ]]>
      </operators>
    </integrate>
  </sequence>

  <output format="hdf5">
    <group>
      <sampling basis="x(&Nsamples;)" initial_sample="yes">
        <moments>density psire psiim trappotential interaction_term</moments>
        <dependencies>wavefunction potentials</dependencies>
        <![CDATA[
          density = mod2(psi);
          psire = psi.Re();
          psiim = psi.Im();
          trappotential = trap;
          interaction_term = U1d*mod2(psi);
```

```
        ]]>
      </sampling>
    </group>
    <group>
      <sampling basis="kx(&Npts;)" initial_sample="yes">
        <moments>fspec</moments>
        <dependencies>wavefunction</dependencies>
        <![CDATA[
          fspec = mod2(psi);
        ]]>
      </sampling>
    </group>
    <group>
      <sampling basis="" initial_sample="yes">
        <moments> atomnumber meanpos deltapos </moments>
        <dependencies> moments </dependencies>
        <![CDATA[
          atomnumber = norm;
          meanpos = expecx/norm;
          deltapos = sqrt(expecxx/norm - (expecx/norm)*(expecx/norm));
        ]]>
      </sampling>
    </group>
  </output>
</simulation>
```