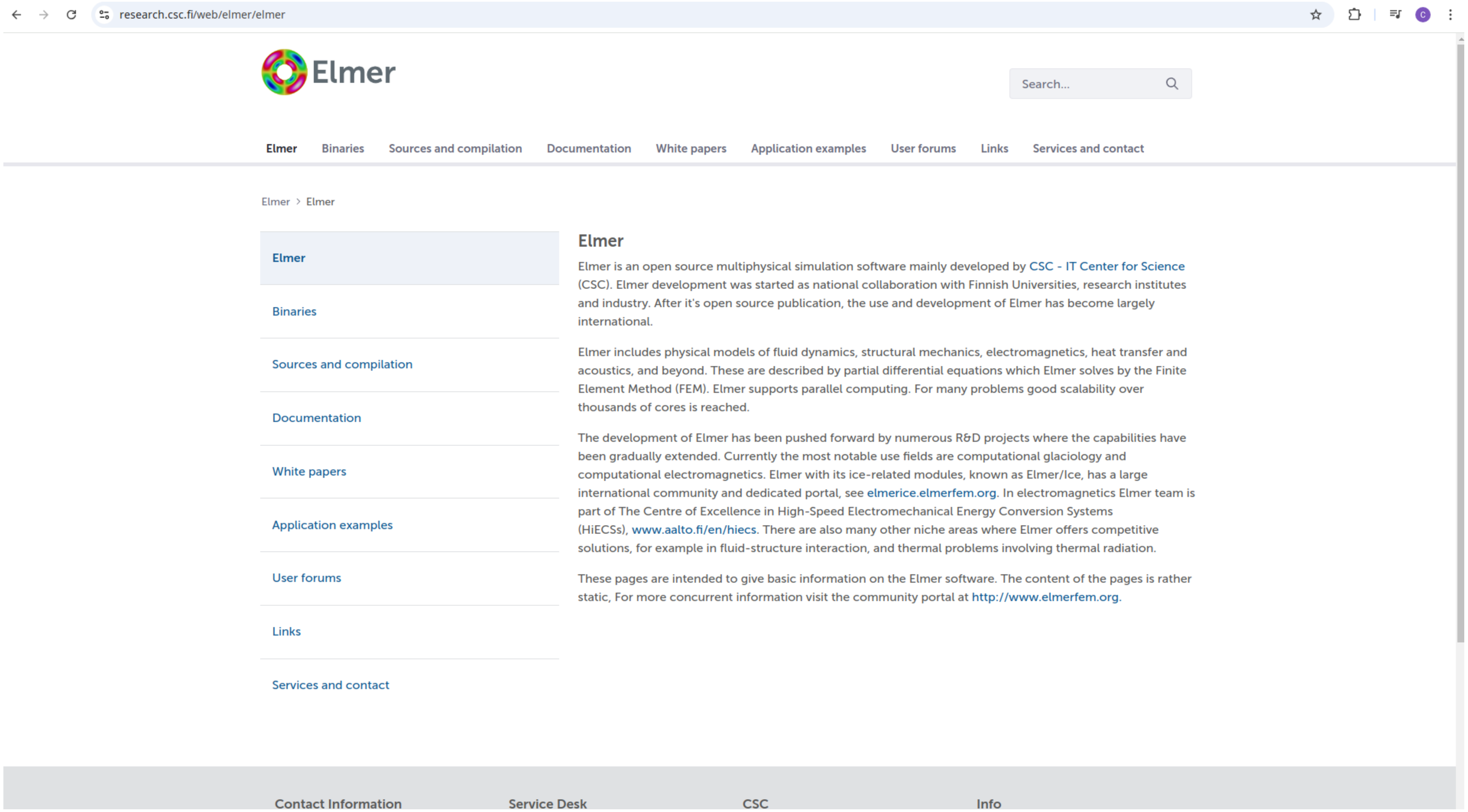
MovingChargeDensity



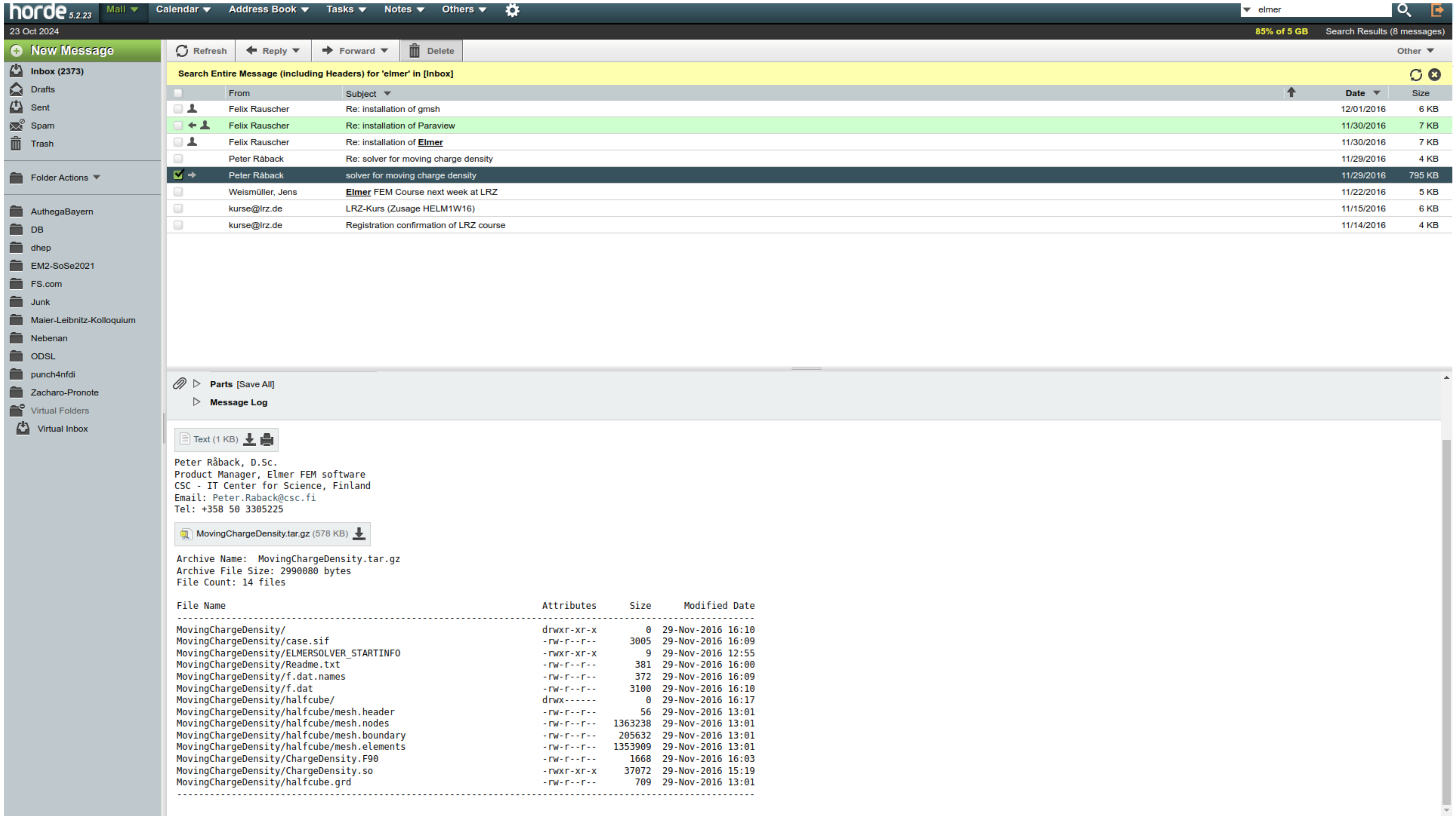
Elmer

Elmer is an open source multiphysical simulation software mainly developed by CSC - IT Center for Science (CSC). Elmer development was started as national collaboration with Finnish Universities, research institutes and industry. After it's open source publication, the use and development of Elmer has become largely international.

Elmer includes physical models of fluid dynamics, structural mechanics, electromagnetics, heat transfer and acoustics, and beyond. These are described by partial differential equations which Elmer solves by the Finite Element Method (FEM). Elmer supports parallel computing. For many problems good scalability over thousands of cores is reached.

The development of Elmer has been pushed forward by numerous R&D projects where the capabilities have been gradually extended. Currently the most notable use fields are computational glaciology and computational electromagnetics. Elmer with its ice-related modules, known as Elmer/Ice, has a large international community and dedicated portal, see elmerice.elmerfem.org. In electromagnetics Elmer team is part of The Centre of Excellence in High-Speed Electromechanical Energy Conversion Systems (HiECSs), www.aalto.fi/en/hiecs. There are also many other niche areas where Elmer offers competitive solutions, for example in fluid-structure interaction, and thermal problems involving thermal radiation.

These pages are intended to give basic information on the Elmer software. The content of the pages is rather static, For more concurrent information visit the community portal at http://www.elmerfem.org.

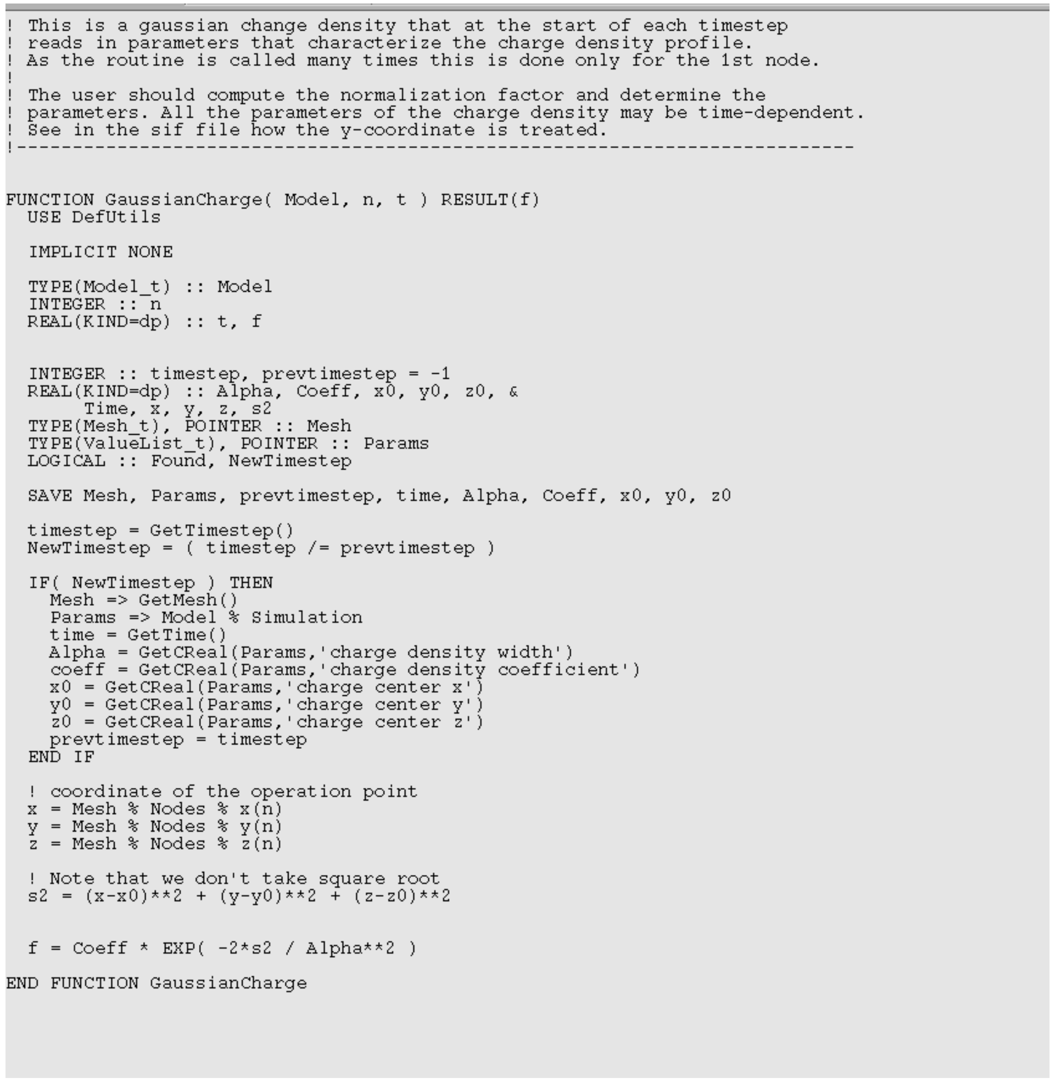


Search Entire Message (including Headers) for 'elmer' in [Inbox]

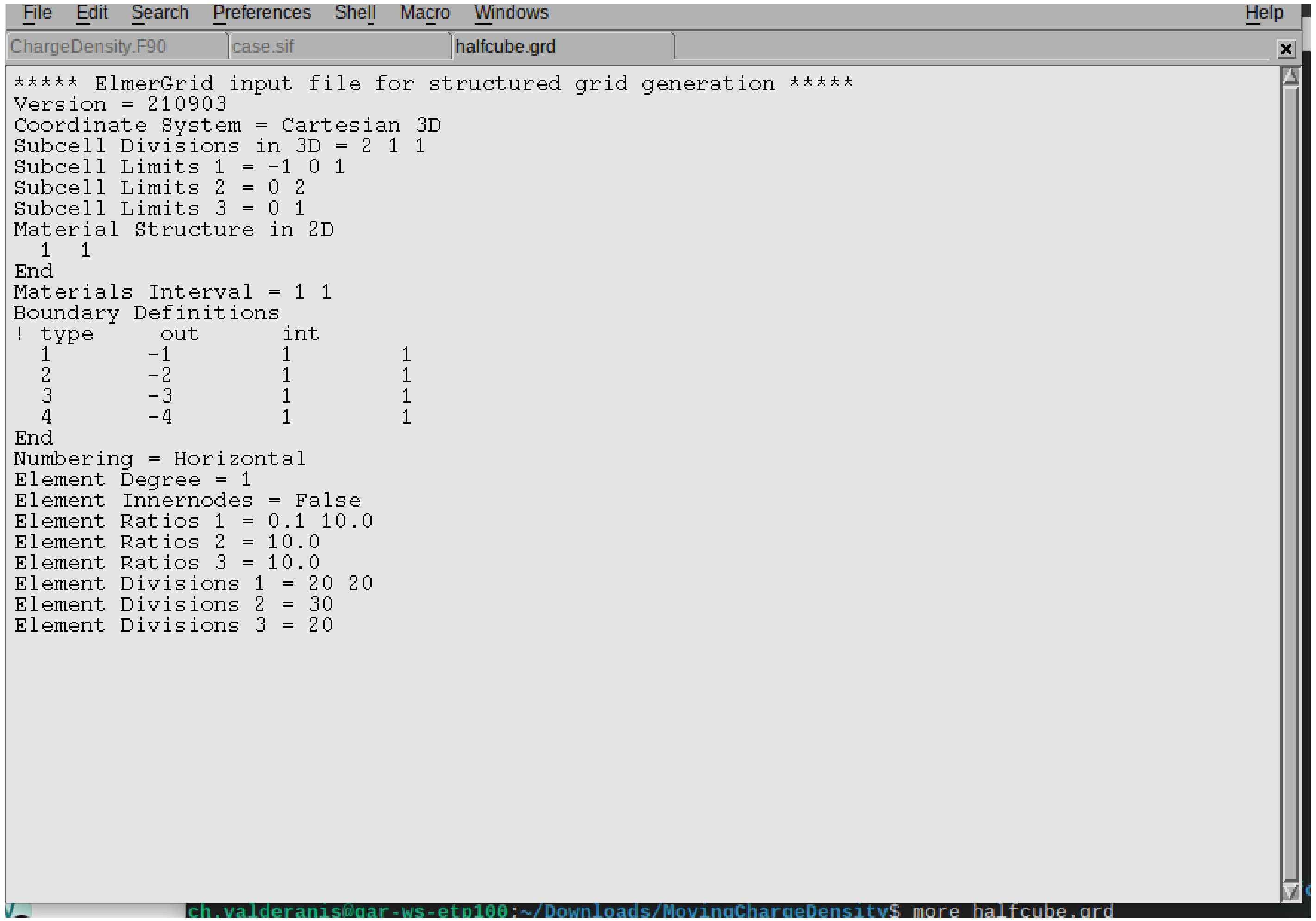
From	Subject	Date	Size
Felix Rauscher	Re: installation of gmsh	12/01/2016	6 KB
Felix Rauscher	Re: installation of Paraview	11/30/2016	7 KB
Felix Rauscher	Re: installation of **Elmer**	11/30/2016	7 KB
Peter Råback	Re: solver for moving charge density	11/29/2016	4 KB
Peter Råback	solver for moving charge density	11/29/2016	795 KB
Weismüller, Jens	**Elmer** FEM Course next week at LRZ	11/22/2016	5 KB
kurse@lrz.de	LRZ-Kurs (Zusage HELM1W16)	11/15/2016	6 KB
kurse@lrz.de	Registration confirmation of LRZ course	11/14/2016	4 KB

```
Peter Råback, D.Sc.
Product Manager, Elmer FEM software
CSC - IT Center for Science, Finland
Email: Peter.Raback@csc.fi
Tel: +358 50 3305225
```

```
Archive Name:  MovingChargeDensity.tar.gz
Archive File Size: 2990080 bytes
File Count: 14 files

File Name                                                        Attributes      Size    Modified Date
------------------------------------------------------------------------------------------------------
MovingChargeDensity/                                             drwxr-xr-x         0  29-Nov-2016 16:10
MovingChargeDensity/case.sif                                     -rw-r--r--      3005  29-Nov-2016 16:09
MovingChargeDensity/ELMERSOLVER_STARTINFO                        -rwxr-xr-x         9  29-Nov-2016 12:55
MovingChargeDensity/Readme.txt                                   -rw-r--r--       381  29-Nov-2016 16:00
MovingChargeDensity/f.dat.names                                  -rw-r--r--       372  29-Nov-2016 16:09
MovingChargeDensity/f.dat                                        -rw-r--r--      3100  29-Nov-2016 16:10
MovingChargeDensity/halfcube/                                    drwx------         0  29-Nov-2016 16:17
MovingChargeDensity/halfcube/mesh.header                         -rw-r--r--        56  29-Nov-2016 13:01
MovingChargeDensity/halfcube/mesh.nodes                          -rw-r--r--   1363238  29-Nov-2016 13:01
MovingChargeDensity/halfcube/mesh.boundary                       -rw-r--r--    205632  29-Nov-2016 13:01
MovingChargeDensity/halfcube/mesh.elements                       -rw-r--r--   1353909  29-Nov-2016 13:01
MovingChargeDensity/ChargeDensity.F90                            -rw-r--r--      1668  29-Nov-2016 16:03
MovingChargeDensity/ChargeDensity.so                             -rwxr-xr-x     37072  29-Nov-2016 15:19
MovingChargeDensity/halfcube.grd                                 -rw-r--r--       709  29-Nov-2016 13:01
------------------------------------------------------------------------------------------------------
```

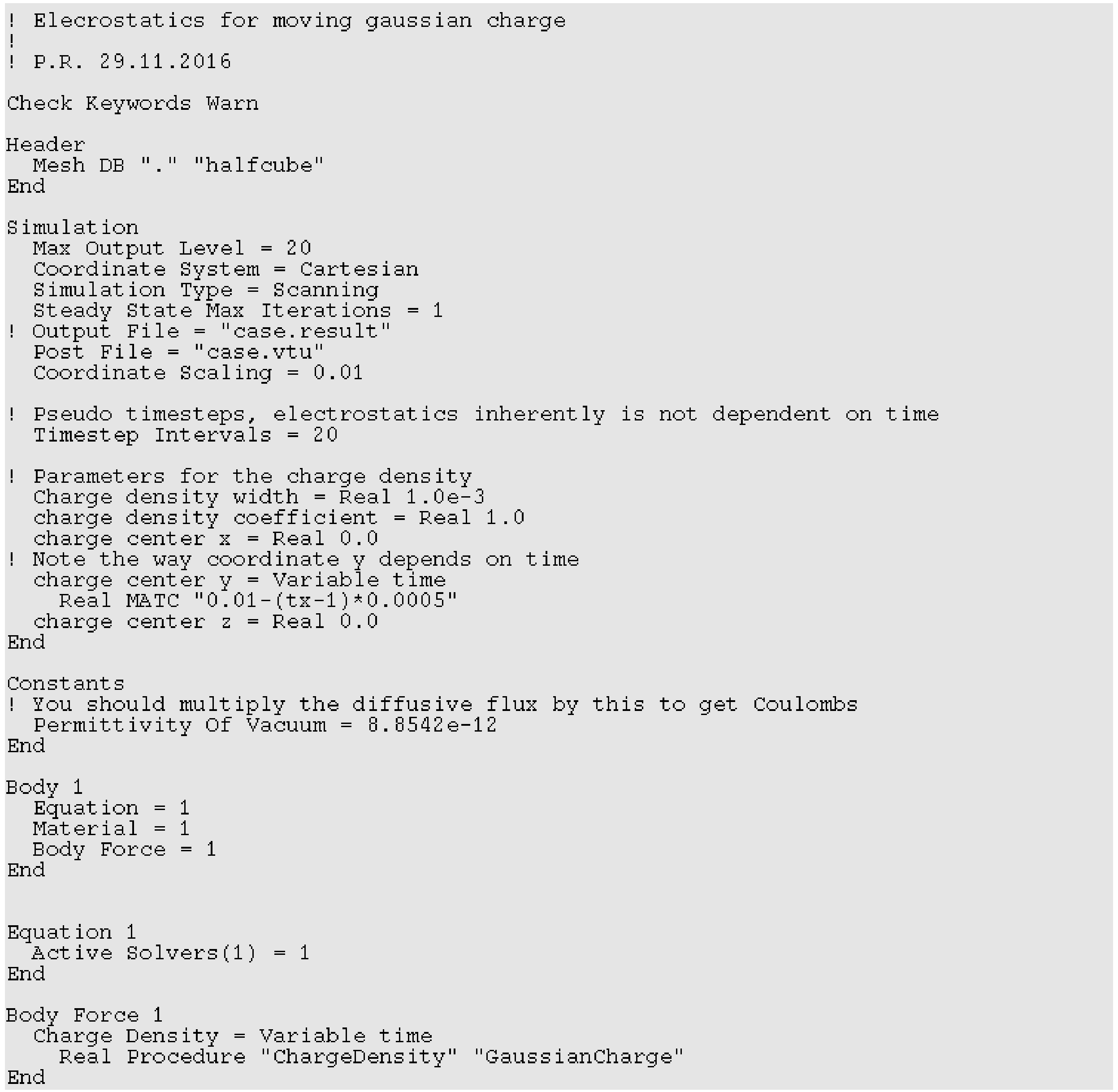


```
! This is a gaussian change density that at the start of each timestep
! reads in parameters that characterize the charge density profile.
! As the routine is called many times this is done only for the 1st node.
!
! The user should compute the normalization factor and determine the
! parameters. All the parameters of the charge density may be time-dependent.
! See in the sif file how the y-coordinate is treated.
!------------------------------------------------------------------------------


FUNCTION GaussianCharge( Model, n, t ) RESULT(f)
  USE DefUtils

  IMPLICIT NONE

  TYPE(Model_t) :: Model
  INTEGER :: n
  REAL(KIND=dp) :: t, f


  INTEGER :: timestep, prevtimestep = -1
  REAL(KIND=dp) :: Alpha, Coeff, x0, y0, z0, &
       Time, x, y, z, s2
  TYPE(Mesh_t), POINTER :: Mesh
  TYPE(ValueList_t), POINTER :: Params
  LOGICAL :: Found, NewTimestep

  SAVE Mesh, Params, prevtimestep, time, Alpha, Coeff, x0, y0, z0

  timestep = GetTimestep()
  NewTimestep = ( timestep /= prevtimestep )

  IF( NewTimestep ) THEN
    Mesh => GetMesh()
    Params => Model % Simulation
    time = GetTime()
    Alpha = GetCReal(Params,'charge density width')
    coeff = GetCReal(Params,'charge density coefficient')
    x0 = GetCReal(Params,'charge center x')
    y0 = GetCReal(Params,'charge center y')
    z0 = GetCReal(Params,'charge center z')
    prevtimestep = timestep
  END IF

  ! coordinate of the operation point
  x = Mesh % Nodes % x(n)
  y = Mesh % Nodes % y(n)
  z = Mesh % Nodes % z(n)

  ! Note that we don't take square root
  s2 = (x-x0)**2 + (y-y0)**2 + (z-z0)**2


  f = Coeff * EXP( -2*s2 / Alpha**2 )

END FUNCTION GaussianCharge
```

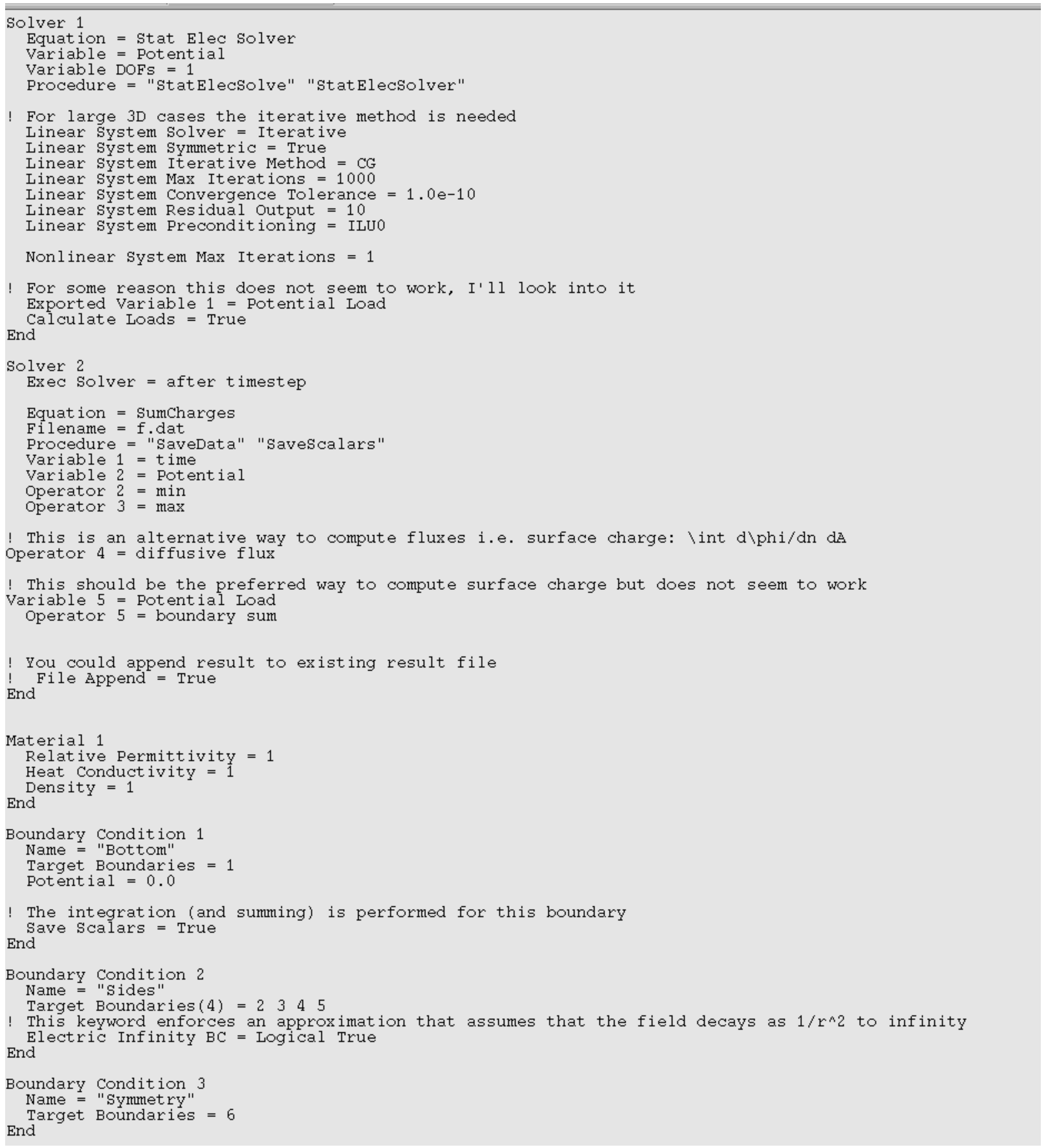


```
***** ElmerGrid input file for structured grid generation *****
Version = 210903
Coordinate System = Cartesian 3D
Subcell Divisions in 3D = 2 1 1
Subcell Limits 1 = -1 0 1
Subcell Limits 2 = 0 2
Subcell Limits 3 = 0 1
Material Structure in 2D
  1  1
End
Materials Interval = 1 1
Boundary Definitions
! type     out      int
  1       -1        1         1
  2       -2        1         1
  3       -3        1         1
  4       -4        1         1
End
Numbering = Horizontal
Element Degree = 1
Element Innernodes = False
Element Ratios 1 = 0.1 10.0
Element Ratios 2 = 10.0
Element Ratios 3 = 10.0
Element Divisions 1 = 20 20
Element Divisions 2 = 30
Element Divisions 3 = 20
```

```
ch.valderanis@gar-ws-etp100:~/Downloads/MovingChargeDensity$ more halfcube.grd
```



```
! Elecrostatics for moving gaussian charge
!
! P.R. 29.11.2016

Check Keywords Warn

Header
  Mesh DB "." "halfcube"
End

Simulation
  Max Output Level = 20
  Coordinate System = Cartesian
  Simulation Type = Scanning
  Steady State Max Iterations = 1
! Output File = "case.result"
  Post File = "case.vtu"
  Coordinate Scaling = 0.01

! Pseudo timesteps, electrostatics inherently is not dependent on time
  Timestep Intervals = 20

! Parameters for the charge density
  Charge density width = Real 1.0e-3
  charge density coefficient = Real 1.0
  charge center x = Real 0.0
! Note the way coordinate y depends on time
  charge center y = Variable time
    Real MATC "0.01-(tx-1)*0.0005"
  charge center z = Real 0.0
End

Constants
! You should multiply the diffusive flux by this to get Coulombs
  Permittivity Of Vacuum = 8.8542e-12
End

Body 1
  Equation = 1
  Material = 1
  Body Force = 1
End


Equation 1
  Active Solvers(1) = 1
End

Body Force 1
  Charge Density = Variable time
    Real Procedure "ChargeDensity" "GaussianCharge"
End
```



```
Solver 1
  Equation = Stat Elec Solver
  Variable = Potential
  Variable DOFs = 1
  Procedure = "StatElecSolve" "StatElecSolver"

! For large 3D cases the iterative method is needed
  Linear System Solver = Iterative
  Linear System Symmetric = True
  Linear System Iterative Method = CG
  Linear System Max Iterations = 1000
  Linear System Convergence Tolerance = 1.0e-10
  Linear System Residual Output = 10
  Linear System Preconditioning = ILU0

  Nonlinear System Max Iterations = 1

! For some reason this does not seem to work, I'll look into it
  Exported Variable 1 = Potential Load
  Calculate Loads = True
End

Solver 2
  Exec Solver = after timestep

  Equation = SumCharges
  Filename = f.dat
  Procedure = "SaveData" "SaveScalars"
  Variable 1 = time
  Variable 2 = Potential
  Operator 2 = min
  Operator 3 = max

! This is an alternative way to compute fluxes i.e. surface charge: \int d\phi/dn dA
Operator 4 = diffusive flux

! This should be the preferred way to compute surface charge but does not seem to work
Variable 5 = Potential Load
  Operator 5 = boundary sum


! You could append result to existing result file
!  File Append = True
End


Material 1
  Relative Permittivity = 1
  Heat Conductivity = 1
  Density = 1
End

Boundary Condition 1
  Name = "Bottom"
  Target Boundaries = 1
  Potential = 0.0

! The integration (and summing) is performed for this boundary
  Save Scalars = True
End

Boundary Condition 2
  Name = "Sides"
  Target Boundaries(4) = 2 3 4 5
! This keyword enforces an approximation that assumes that the field decays as 1/r^2 to infinity
  Electric Infinity BC = Logical True
End

Boundary Condition 3
  Name = "Symmetry"
  Target Boundaries = 6
End
```