SLang - the Next Generation

Tutorial

Christian Bucher, Sebastian Wolff
Center of Mechanics and Structural Dynamics
Vienna University of Technology

November 8, 2010

0.1 Random fields on an FE mesh

A triangle finite element mesh as generated by **gmsh** is imported to *SLangTNG*. Then a nodal random field $F(x, y, z)$ is defined. Its correlation function is assumed to be isotropic exponential

$$R_{FF}(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(\frac{||\mathbf{x}_1 - \mathbf{x}_2||}{L_c}\right) \tag{1}$$

with a correlation length $L_c = 0.2$. The field is assumed to be Gaussian. The discrete Karhunen-Loeve expansion of the random field required the computation of the eigenvalues λ_k and eigenvectors ϕ_k of the correlation matrix. Here the $N = 100$ largest eigenvalues and corresponding eigenvectors are computed. The a Monte Carlo simulation of the random field is carried out. The *SLangTNG*-code to solve this problem is given below.

```
--[[
SLangTNG
Simple test example for random fields
(c) 2009 Christian Bucher, CMSD-VUT
--]]

-- Import triangular mesh created by gmsh
  struct=tngfem.TNGStructureImportGmsh("panel.msh")
  nd=struct:GlobalDof()

-- Define section and material properties (Gmsh provides only the mesh)
  ss=struct:AddSection(301, "SHELL", 0, 0.01)
  ss:SetColor(0,200,200,255)
  struct:SetSection(301)

-- Define a random field for nodal properties, the correlation function is
-- exponential, the distribution type is normal
  field=tngfem.TNGRanfield(struct, "NODES", "EXPONENTIAL", "LOGNORMAL")

-- Define mean value
  field:SetMean(.1);

-- Define standard deviation
  field:SetSigma(.03);

-- Define correlation length
  field:SetCorrelationLength(.5);

-- Assemble the correlation matrix
  corr=field:GetSparseCorrelation();

-- Perform the Karhunen-Loeve decomposition (Eigenvalue analysis)
  N=100
  val, vec = corr:EigenLargest(N);
  print("val", val)
  print("vec", vec)

-- Prepare visualization of the eigenvectors
  alldisp=struct:GetAllDisplacements()
  super=tnggraphics.TNGSuperVisualize(50, 50, 1000, 800, "Imperfection shapes")

-- Loop showing some eigenvectors interpreted as z-displacements of all nodes
  for i=0,3 do
    shape=vec:Col(N-1-i*2)
-- Normalize shape zu maximum value of 1
    shape=shape/shape:MaxCoeff()
    alldisp:SetCols(shape, 2)
    newcolumn = math.mod(i,2)==0
-- Assign diplacements for visualization and draw deformed structure
    struct:SetAllDisplacements(alldisp)
    v = super:AddVisualize("Shape "..i*2, newcolumn)
    v:Perspective(true)
    v:Lighting(true)
    v:SetAngles(50,30,0)
    v:Draw(struct, .1)
    end

-- Monte Carlo simulation, start with standard Gaussian variables
  NSIM = 30
  random = stoch.Simulate(N, NSIM)
```

```
  for i=0,3 do
    s=random:GetCols(i)
-- Produce one sample of the lognormal field
    sample=field:Sample(s, val, vec)
    alldisp:SetCols(sample, 2)
    newcolumn = math.mod(i,2)==0

-- Assign diplacements for visualization and draw deformed structure
    struct:SetAllDisplacements(alldisp)
    v = super:AddVisualize("Sample "..i, newcolumn)
    v:Perspective(true)
    v:Lighting(true)
    v:SetAngles(50,-30,0)
    v:Draw(struct,1)
    end

-- Output graphics
  super:File("shapes.pdf")
```

The resulting eigenvectors as well as the Monte Carlo samples are shown in Fig. ??.

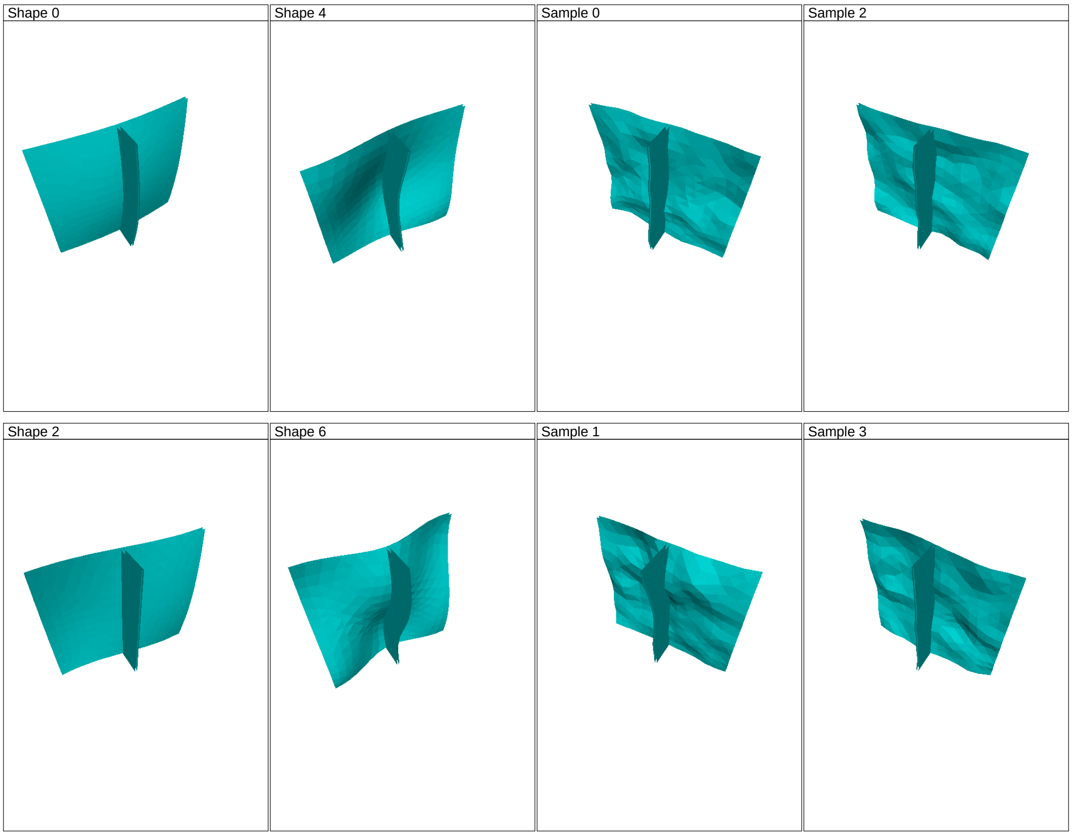

Figure 1: Random field on a triangle mesh