Modifications of OpenSees to Further Enable Reliability/Sensitivity/Optimization Technologies

Prof. Michael H. Scott (OSU) and
Prof. Kevin R. Mackie (UCF)

PEER Annual Meeting
Hotel Shattuck Plaza
Berkeley, CA

October 1, 2011

Project Objectives

1. Support PEER research in probabilistic analysis of building and transportation systems
2. Improve usability of reliability and sensitivity modules
   - More flexibility in creating uncertainty models
   - Obtain uncertainty analysis results in free-form
3. Software redesign of uncertainty modeling framework
   - Reduce tight coupling of classes and code duplication
   - Profiling to identify computational bottlenecks
   - Extend framework beyond familiar application domains
4. Consistency with other OpenSees developers
   - SNOPT optimization at UCSD
   - TELM reliability at UCB

Generalization of Parameter Class (*In progress*)

- Accommodate parameters beyond FE model, its response, and uncertainty variables
- Compute derivatives with respect to:
  1. FE model and its response (completed)
  2. Random and design variable parameters (completed)
  3. Uncertainty model and its response (in progress)
  4. User-defined scripting language parameters (in progress)
- Each subclass implements getSensitivity() method as indirection toward methods for given uncertainty analysis
  - FEModelParameter – returns 0 or 1
  - FEResponseParameter – calls Node or Element class to get sensitivity

Improvements in Usability (*Completed, in SVN*)

1. Allow non-sequential tagging of uncertainty objects
   - Parameters, random variables, positioners, performance functions
   - Free-form approach for uncertainty analysis of large models
   - Tcl commands return lists of tags for script-level data management
2. Identify and update parameters within a Tcl script
   - Easy approach to Monte Carlo simulation
   - Finite difference sensitivity without wiping out entire model
   - Avenues in to "black box" for third party FE model updating
3. Rethink how uncertainty analysis results are given to user
   - Fixed-format text output file –> house of cards
   - Tcl commands, e.g., `[getBetaFORM $pfTag]` will allow user to create files with customized format

Improvements in Usability (*Completed, in SVN*)

4. Use Tcl API to evaluate performance functions
   - Parsing of specialized syntax was difficult to maintain/extend `performanceFunction 2 ‘‘0.2 - u_2_1’’` (*Now deprecated*)
   - Use Tcl's inherent variable substitution and procedure calls (user-defined or OpenSees-defined) `performanceFunction 2 ‘‘0.2 - \[nodeDisp 2 1\]’’`
5. User-defined gradients of performance functions
   - Reliance on finite differences even when analytic gradients of FE response
   - Simple performance functions: use OpenSees/Tcl commands `gradPerformanceFunction 2 $rvTag ‘‘-\[sensNodeDisp 2 1 $paramTag\]’’`

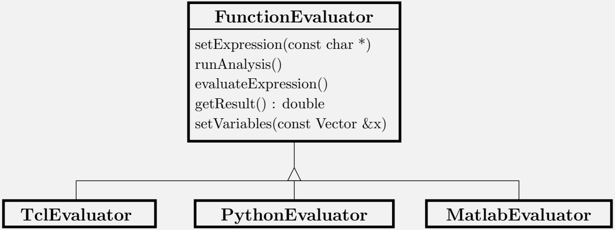
Changes to Random Variable Constructors (*Completed, not in SVN*)

- Only two constructors for each random variable
  - `RandomVariable(int tag, double mean, double stdev)`
  - `RandomVariable(int tag, const Vector &params)`

  Each subclass figures out how to convert mean/stdev to distribution parameters and is responsible for interpreting Vector of parameters
- Previously four constructors for each random variable, some of which went unused (dead code)
- Removed about 500 lines of code from TclReliabilityBuilder.cpp
- Removed manual entry of π and Euler-Γ
- Removed inter-dependency of RVs, e.g., Lognormal RV used instance of Normal RV as a member variable

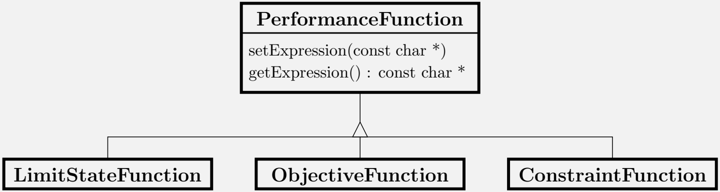
Abstract FunctionEvaluator Class (*Completed, not in SVN*)

- Generalization of previous GFunEvaluator hierarchy
- All parsing pushed to APIs of subclass scripting languages
- Any reliability/sensitivity/optimization component can call FunctionEvaluator (gradient, limit state function, analytic gradient, etc.)
- setVariables() method creates arrays in local namespace, e.g., Tcl $par($i)



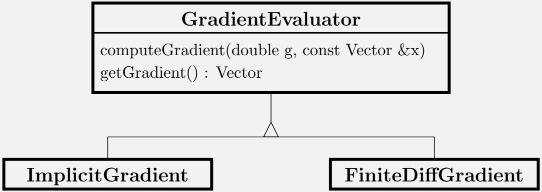
Abstract PerformanceFunction Class (*Completed, not in SVN*)

- Simplification of previous LimitStateFunction hierarchy
- Only stores expression to be parsed by scripting language according to concrete subclasses of FunctionEvaluator
- No longer stores results of various uncertainty analyses
- Stripped out all explicit C++ parsing
- Subclasses have specific behavior, e.g., LimitStateFunction stores analytic expressions for gradient of the limit state function

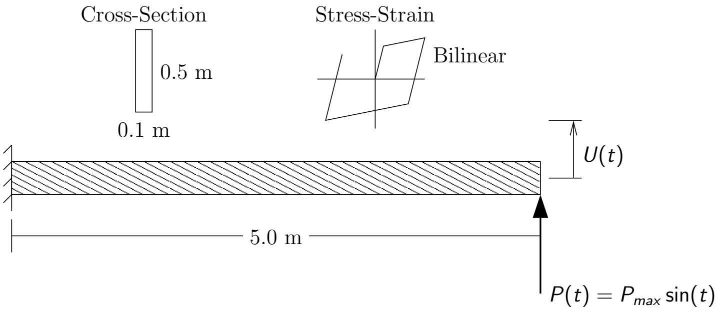


Abstract GradientEvaluator Class (*In progress*)

- Generalization of previous GradGEvaluator hierarchy
- Stripped out all C++ parsing; now calls instance of FunctionEvaluator
- Can compute gradients of any of four types of parameters
- Implicit gradient depends only on Parameter->getSensitivity(), so works with FEA outputs, user functions, etc. listed above
- Finite difference gradient makes repeated calls to FunctionEvaluator

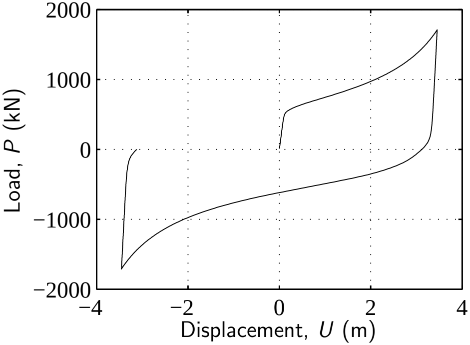


Example – Material and Geometric Nonlinear Cantilever



- Discretize with five frame finite elements (linear curvature, constant axial deformation approximation)
- Corotational formulation for large displacements
- Numerically integrate stress-strain response over cross-section
- $E = 2.0e8$ kPa, $\sigma_y = 4.1e5$ kPa, 2% kinematic strain-hardening
- One load cycle with $P_{max} = 1710$ kN (5 times yield load)

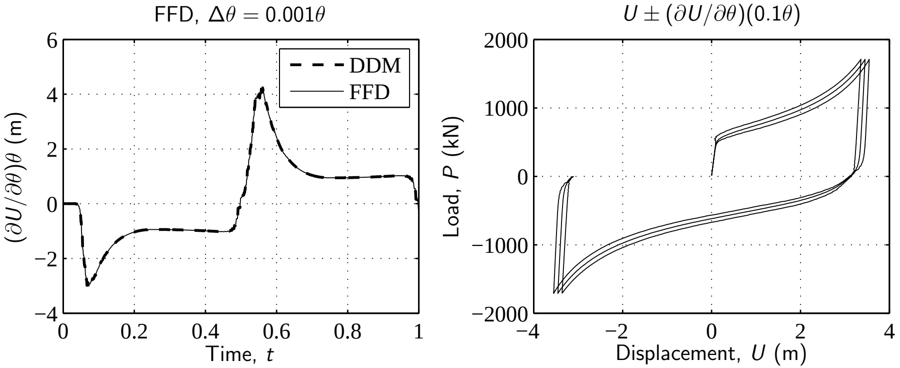
Mean Load-Displacement Response



- Material yield at about 400 kN load
- Tension stiffening at about 1 m displacement
- Elastic unloading and reverse cyclic yielding

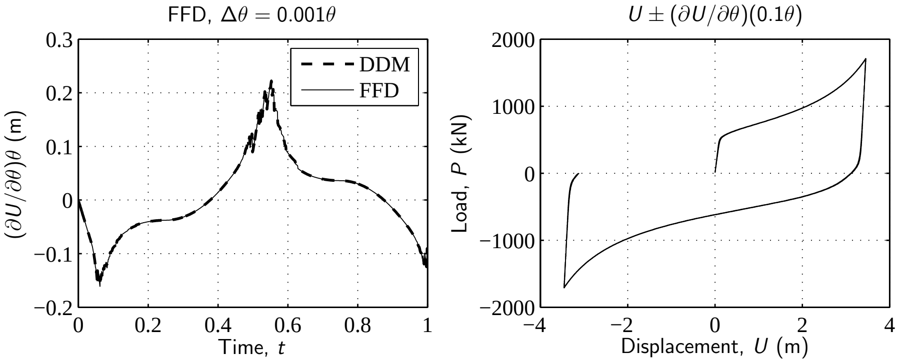
Sensitivity to Yield Stress

DDM verification and response envelope for strength parameter



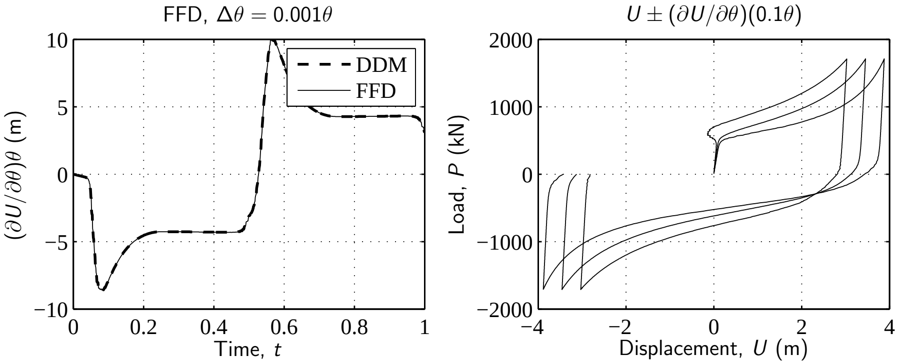
Sensitivity to Elastic Modulus

DDM verification and response envelope for stiffness parameter



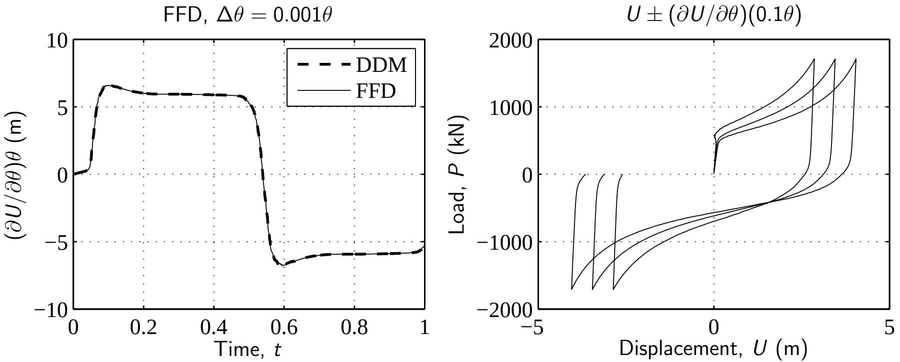
Sensitivity to Cross-Section Depth

DDM verification and response envelope for local geometric parameter



Sensitivity to Cantilever Length

DDM verification and response envelope for global geometric parameter

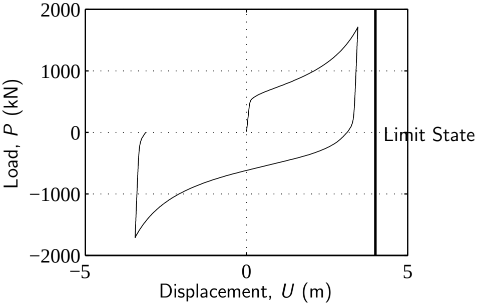


Uncertainty Analysis

Determine probability that peak displacement exceeds 4.0 m

$$g(\mathbf{x}) = 4.0 - U$$

$X_1 \sim LN(4.1e5, 0.05)$ MPa maps to yield stress, σ_y
$X_2 \sim LN(2.0e8, 0.1)$ MPa maps to elastic modulus, E
$X_3 \sim N(1710, 0.15)$ kN maps to applied load, P_{max}
$X_4 \sim N(5.0, 0.02)$ m maps to cantilever length, L
$X_5 \sim N(0.5, 0.04)$ m maps to cross-section depth, d



FORM Results

HLRF converges in 7 iterations

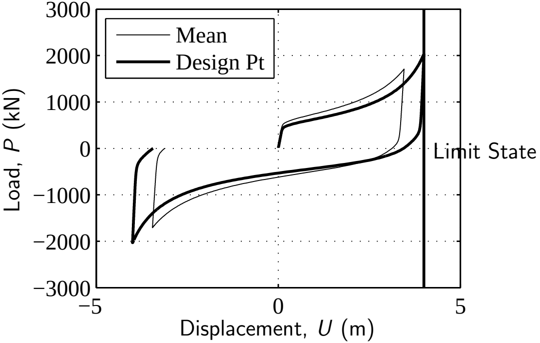
Reliability index, $\beta = 1.983$

Probability of failure, $p_f = \Phi(-1.983) = 2.368\%$

Random Variable, i		μ_i	X_i^*	α_i
1	σ_y	4.1e5 kPa	3.897e5 kPa	-0.2317
2	E	2.0e8 kPa	1.996e8 kPa	-0.005413
3	P_{max}	1710 kN	2032 kN	0.6393
4	L	5.0 m	5.100 m	0.5008
5	d	0.5 m	0.4787 m	-0.5356

Mean and Design Point Response

Cantilever load-displacement response at realization of random variables in $\mathbf{x}^*$



Multiple Hazards

Simultaneous *natural* hazards

- Extremely rare
- Effects unknown due to lack of data
- Prohibitively expensive for design

Simultaneous man-made and natural hazards less rare

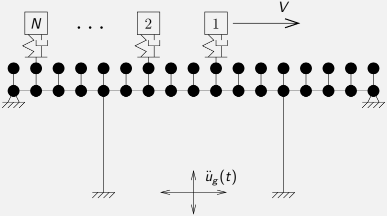
- LRFD load combinations
- Uniform level of safety

Computational challenges

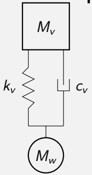
- Performance-based framework
- Quantification of uncertainty
- Time history analysis

Modeling Approach

- Zero-length container elements connected to girder nodes
- Sweep axles across girder at constant velocity
- Apply uniform horizontal and vertical ground acceleration



Axle spring-mass-damper system



M_v – vehicle mass on axle
k_v – suspension stiffness
c_v – suspension damping
M_w – wheel mass

Reliability Analysis

- Perform FORM analysis at each time step
- Limit state on girder moment-shear interaction

$$g(\mathbf{x}) = 1 - \left|\frac{M}{M_n}\right|^3 - \left|\frac{V}{V_n}\right|^3$$

M, V – moment, shear demand from FEA

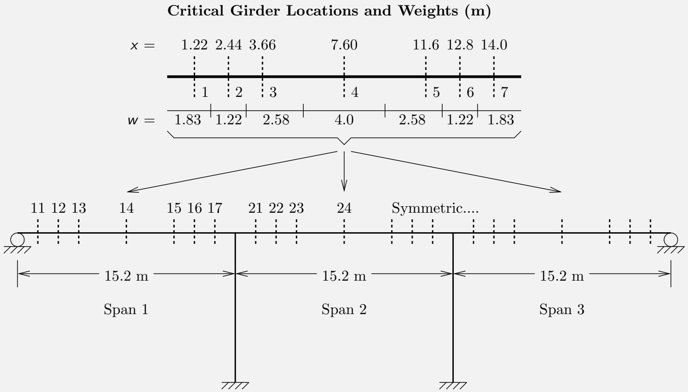
M_n, V_n – nominal moment, shear capacity

$\mathbf{x}$ – vector of random variables mapped to FE domain

- Use analytic derivatives of FE response and limit state function
  - FORM search directions
  - Importance and sensitivity measures
  - Not using finite differences

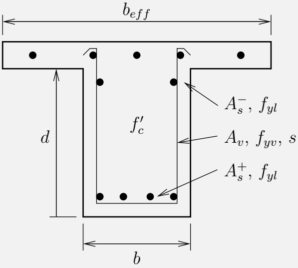
Example Bridge

Conventionally reinforced concrete deck-girder bridge

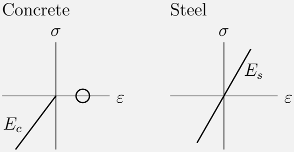


- Limit state function on M-V interaction of section 17

Girder Reinforcing Details



Section Label	d (cm)	b (cm)	A_s^- (cm^2)	A_s^+ (cm^2)	s (cm)
11	122	33	2.58	46.1	30.5
12	122	33	2.58	60.4	48.3
13,14	122	33	2.58	80.5	48.3
15	122	33	20.3	47.4	23.0
16	122	39	40.3	30.2	23.0
17,21	122	45	60.4	30.2	23.0
22	122	39	55.0	30.2	23.0
23	122	33	40.3	30.2	30.5
24	122	33	2.58	46.6	48.3



$$f'_c = 22.8 \text{ MPa}$$
$$f_{yl} = 276 \text{ MPa}$$
$$f_{yv} = 276 \text{ MPa}$$
$$A_v = 2.58 \text{ cm}^2$$
$$E_c = 22.6 \text{ GPa}$$
$$E_s = 200 \text{ GPa}$$

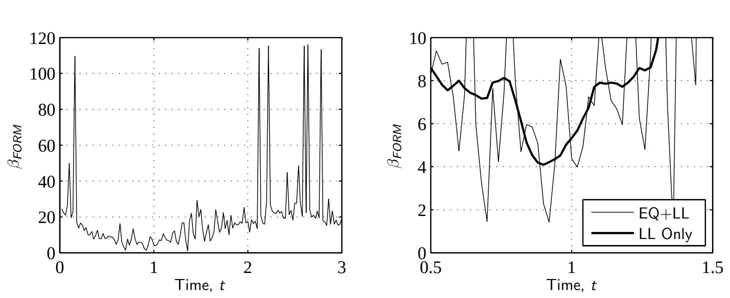
- Using fiber-discretized cross-sections so that sensitivity to reinforcing details is computed

Uncertainty Modeling

Random Variable, i		Mean, μ_i	COV, δ_i	Distribution
1	d	122 cm	0.015	Normal
2	b	45 cm	0.015	Normal
3	A_{top}	27.4 cm^2	0.024	Normal
4	A_{bot}	30.2 cm^2	0.024	Normal
5	E_c	22.6 GPa	0.08	Lognormal
6	G_c	9.41 GPa	0.08	Lognormal
7	E_s	200 GPa	0.06	Lognormal
8	P_1	53.4 kN	0.2	Lognormal
9-15	$P_2 - P_8$	95.6 kN	0.2	Lognormal

- Random variables 1-7 map only to section of interest

Reliability – Combined Earthquake and Live Load



- Minimum reliability index: $\beta = 1.429$ ($p_f = 0.07650$)
- Earthquake only (not shown)
  - Negligible probability of failure for the girder moment-shear limit state function
  - Evidenced as β increases when truck exits bridge (for EQ+LL shown)

Remaining Tasks

1. Overhaul of uncertainty analysis
   - Continue to strip classes down to basic functionality
   - Derive FORM, SORM, TELM, etc. from common base class
2. Concurrent development of TELM and SNOPT within framework
3. External recorders to store uncertainty analysis results
4. Continued documentation online and presentations at OpenSees workshops