Accelerating Mathematical Knot Simulations
with R on the Web

Juan Lin¹,³, Di Zhong², Yiwen Zhong³, Hui Zhang¹,

¹ University of Louisville, Louisville, KY, USA
² Indiana University, Bloomington, IN, USA
³ Fujian Agricultural and Forestry University, Fujian, China

December 12, 2016
Accelerating Mathematical Knot Simulations with R on the Web

Content

Introduction

A New Visualization Paradigm for Exploring Mathematical Knots

Acceleration Geometric Deformations of Complexity

MathSimWeb-Putting Computation and Visualization Together

Conclusion
I Introduction

(I) Twist or untwist in either direction.
(II) Move one loop completely over another.
(III) Move a string completely over or under a crossing.

Figure 1. The three Reidemeister moves.
I Introduction

✓ Self-deformable model

Figure 2. Typical screen images of the self-deformation.
Accelerating Mathematical Knot Simulations with R on the Web

## Introduction

- Self-deformable model
- Web-based interface
- Parallelization

Figure 3. Prototype of MathSimWeb.
II A New Visualization Paradigm for Exploring Mathematical Knots

✅ Force-directed Algorithm

→ Force Laws for Automatic Topological Refinement

• Attractive Mechanical Force
  \[ F_m = H r^{1+\beta} \]
  \( r \) is the distance between masses
  \( H \) is a constant

• Repulsive electrical force:
  \[ F_e = K r^{-(2+\alpha)} \]
  \( r \) is the distance between masses
  \( K \) is a constant
II  A New Visualization Paradigm for Exploring Mathematical Knots

✓ Collision Avoidance for Topology Preservation

• Point-segment collision
• Segment-segment collision
II A New Visualization Paradigm for Exploring Mathematical Knots

✓ Adding Masses to Constrain Deformation in Configuration

Figure 4. The topological relaxation of a curve.
Adding Overriding Forces to Supplement Unguided Refinement

Figure 5. Untying an overhand knot with user-defined overriding forces in conjunction with the self-deformation.
III Accelerating Geometric Deformations Of Complexity

- Extracting Key Moments for Mathematical Movies

→ Through Identifying the minima number of crossing points among all possible 2D projections
library(Matrix);
n <- 8192;
X <- Hilbert(n);
A <- nearPD(X);
system.time(B <- chol(A$mat));
  # user    system   elapsed
    97.990   0.356   98.591
library(HiPLARM)
system.time(B <- chol(A$mat));
  # user    system   elapsed
      1.012   0.316   1.337

Figure 3: A brief example to highlight the benefits of using HiPLARM.
IV MathSimWeb — Putting Computation and Visualization Together

System Architecture

• A back-end module

R(HiPLAR)

```r
library(Matrix);
n <- 5000;
system.time(KnotSim(n));
# user system elapsed
# 107.90 0.356 108.59
library(HiPLARM)
system.time(KnotSim(n));
# user system elapsed
# 21.012 0.316 22.34
```

Figure 3. MathSimWeb.
V Conclusion

Figure 3. MathSimWeb.
Accelerating Mathematical Knot Simulations with R on the Web
Accelerating Mathematical Knot Simulations with R on the Web