Unified 4D Trajectory Approach for Integrated Management of Commercial Air and Space Traffic

Federal Aviation Administration

FAA CoE for CST Technical Meeting

Millennium Harvest House, Boulder, CO

November 9, 2011

Juan J. Alonso and Thomas Colvin
Department of Aeronautics & Astronautics
Stanford University

aerospacedesignlab

Overview

- Team members
- Purpose of Task
- Research Methodology
- Results
- Next Steps
- Contact Information

Team Members

- PI: Juan J. Alonso, Department of Aeronautics & Astronautics, Stanford University
- Thomas J. Colvin, Graduate Student, Department of Aeronautics and Astronautics, Stanford University
- Collaborations/discussions with:
  - Banavar Sridhar, NASA Ames
  - Karl Billimoria, NASA Ames

Purpose of Task

- Projected growth in demand will make it increasingly hard to accommodate launches on a SUA basis
- Looking for a more rational approach that:
  - can adapt to fluctuating frequency of launches
  - can accommodate uncertainties in trajectories
  - ensures proper separation at all times
  - can be integrated with FAA's NextGen system

Research Objectives

1. Develop plausible architectures for an Integrated Airspace Management System (IAMS)
2. Research and develop the foundation of such a tool based on time-space probabilistic trajectories
3. Create a prototype implementation for a proof-of-concept system

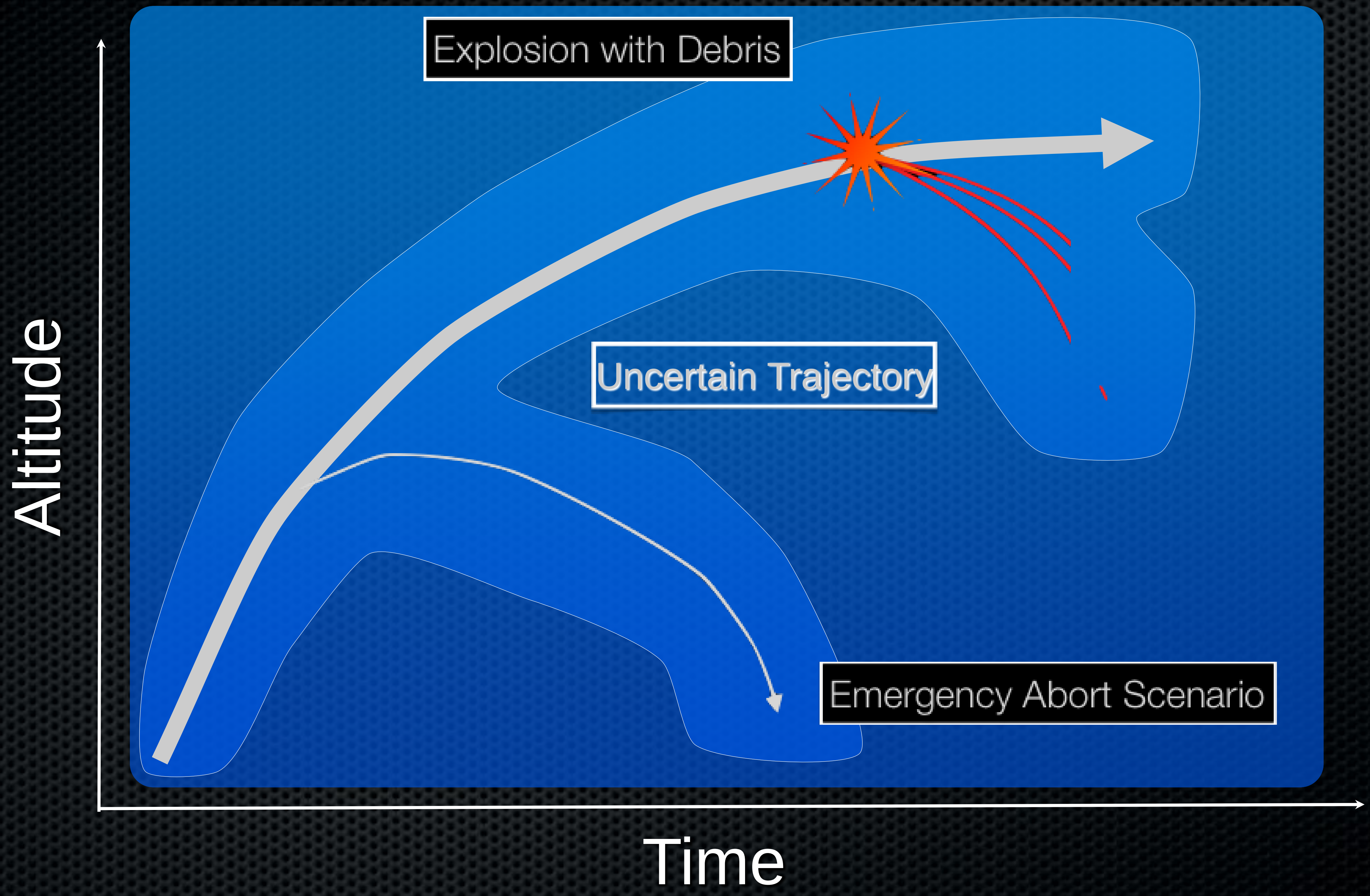
- During first few months, we are focusing on item 2

Methodology & Results

- Problem:
  - Need Special Use Airspace (SUA) for rocket launch
  - Current method for creating SUA may be overly conservative
  - Fairness issues: are we favoring one industry over another?
  - No quantitative framework for creating SUAs

- Proposed Solution:
  - Create a probabilistic framework for creating SUAs to a specified level of safety

Conceptual Framework

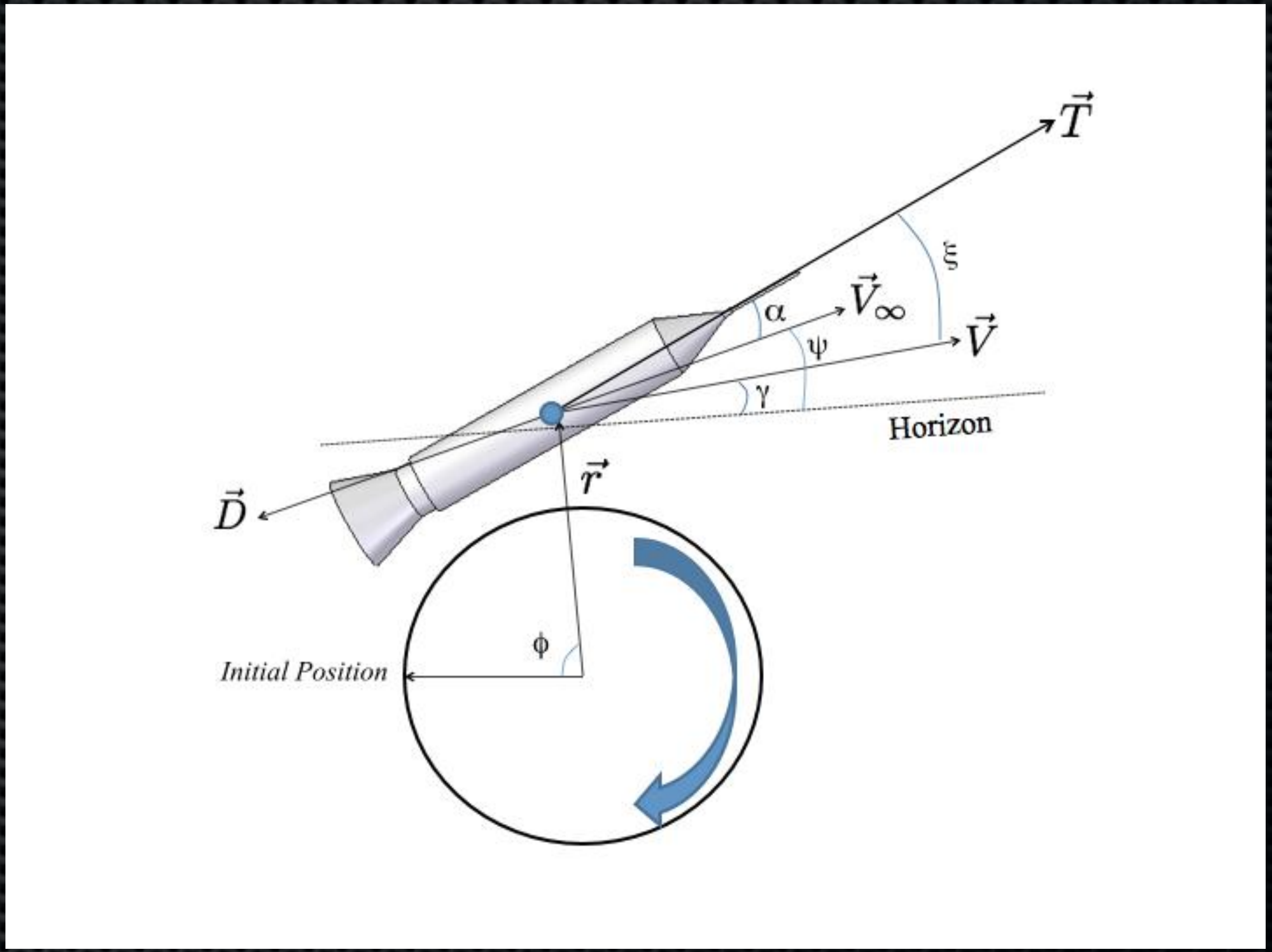


Initial Research Goals

- Focus on:
  - ✓ Investigate ways in which a compact 4-D envelope can be created and specified
  - ✓ Demonstrate the 4-D envelope concept in 3-D (x,y,t)
  - ✓ Begin creating a software architecture that generates 4-D envelopes for specific launch profiles
  - ✓ Use Monte Carlo simulation to <u>approximate</u> the rocket location PDF, sampled at many points, to a given level of safety
- Provide hooks for, but do not spend significant time on (refined later):
  - − Accurate characterization of weather profiles, failure modes and probabilities, debris model

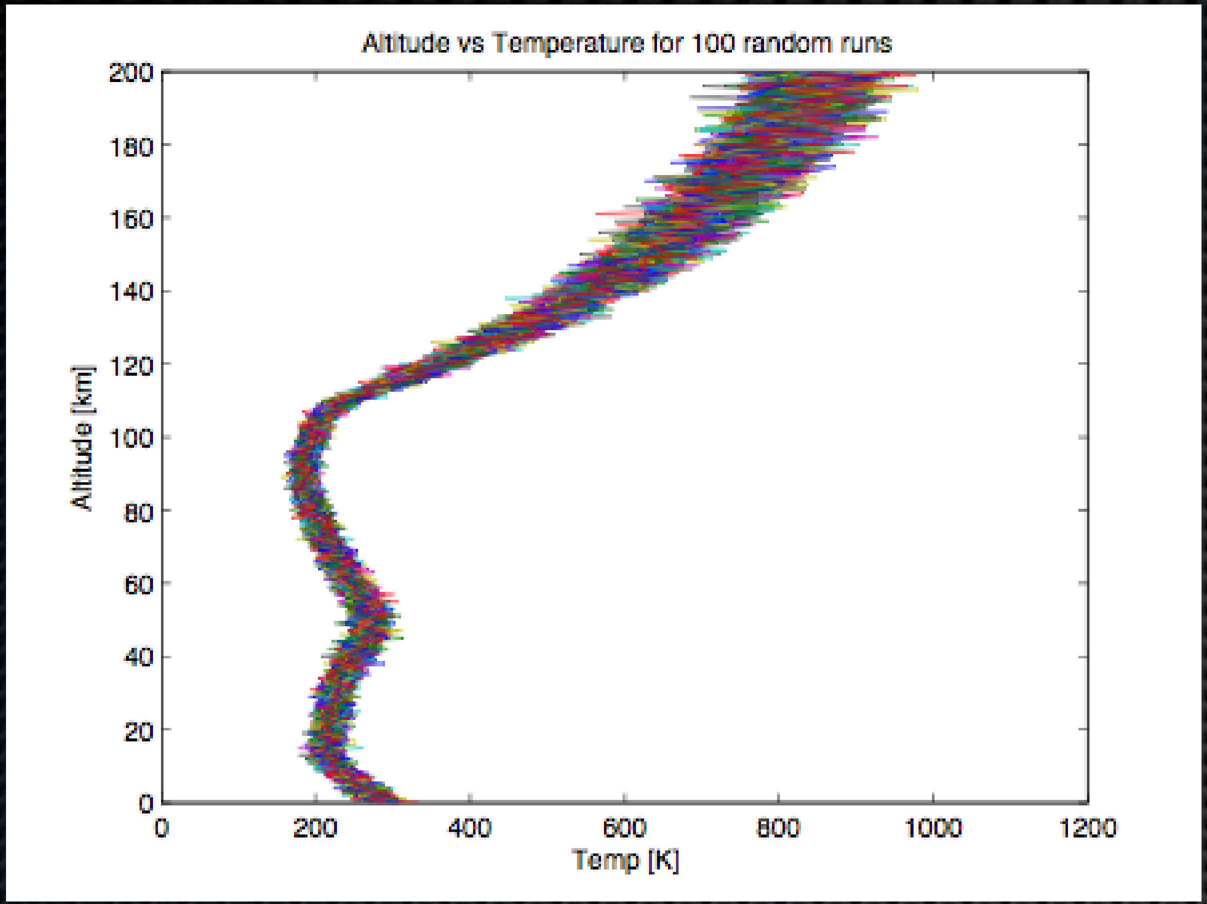
Nominal Trajectory

- 2-D round rotating Earth
  - Propagate r, V, φ, γ
- SSTO launch vehicle
- Optimal trajectory has thrust vectoring (T, ξ)
- Aerodynamic effects are roughly modeled

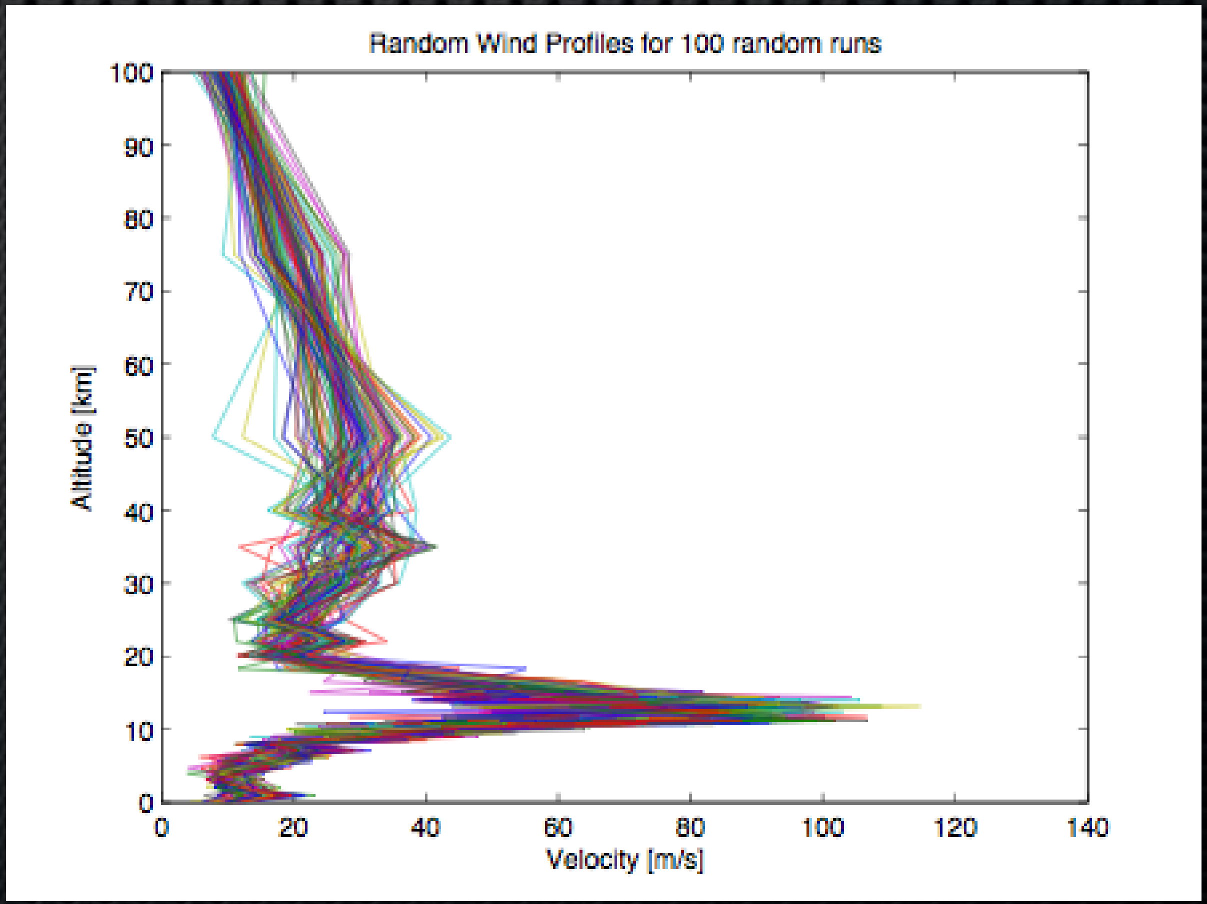


Source: Capristan, F. "Aerodynamic Effects in Launch Vehicle Optimal Trajectories"

Weather Uncertainty



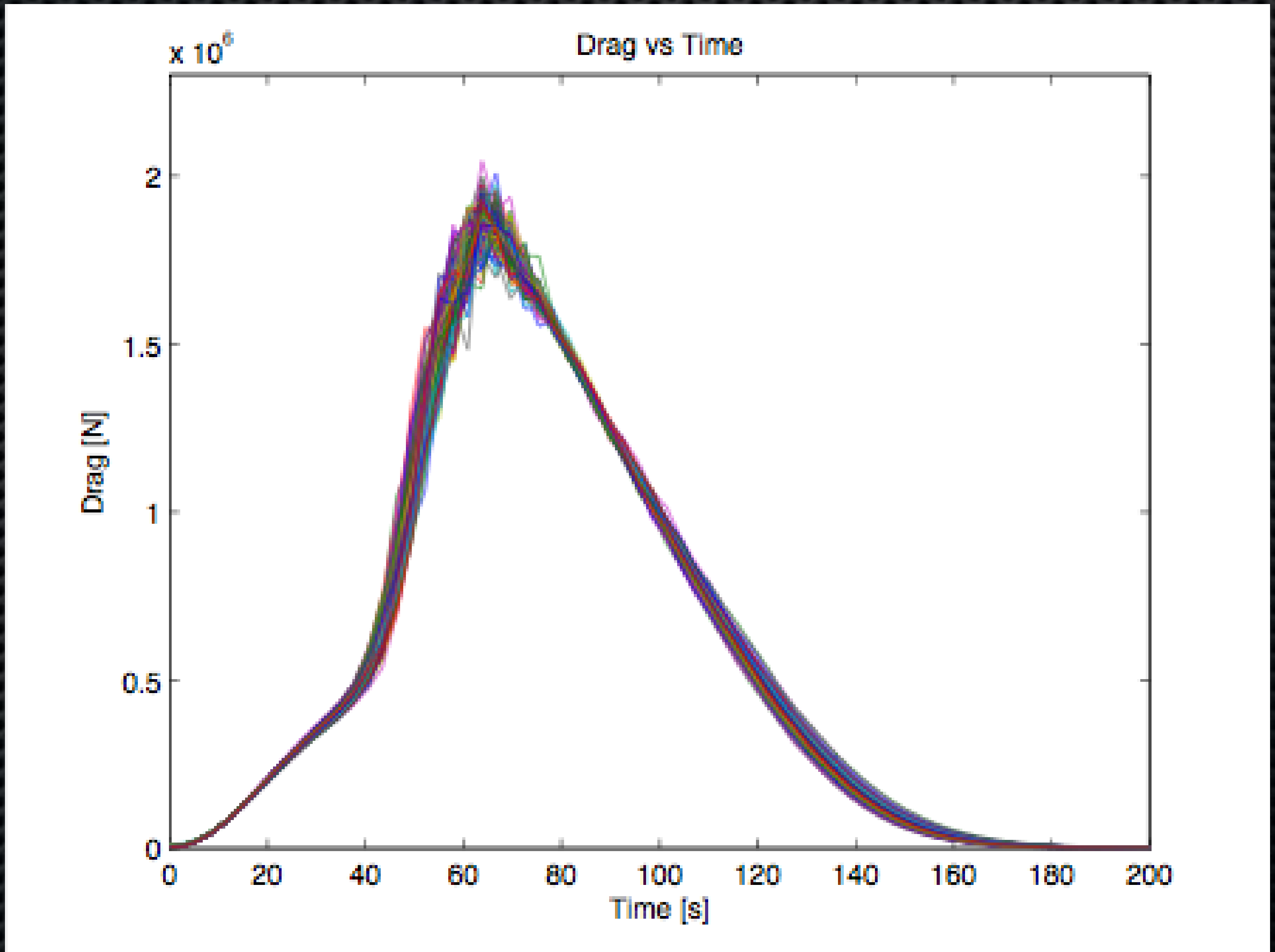
5% Uncertainty
in Temperature



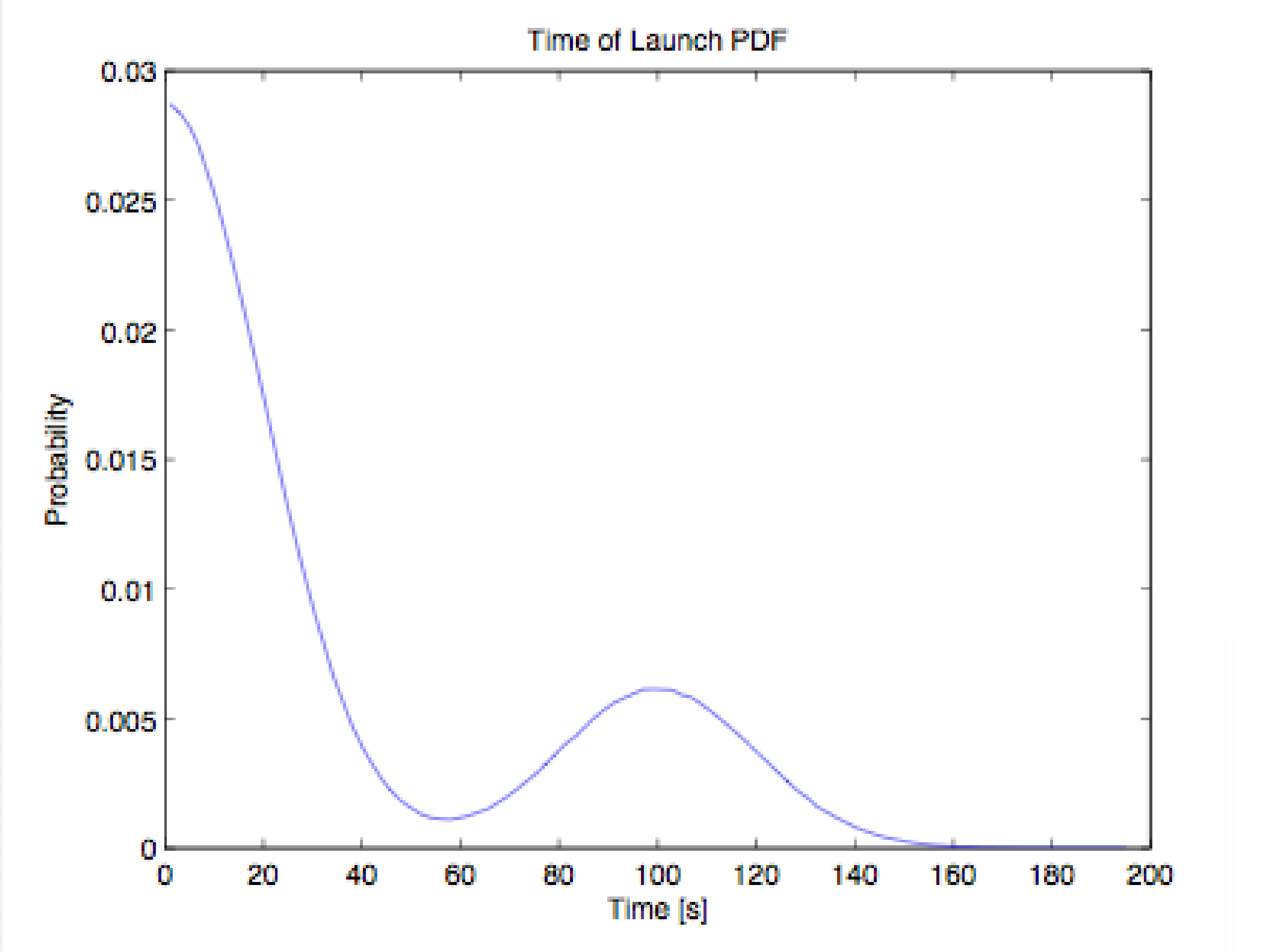
20% Uncertainty
in Wind Velocity

Creates Drag Uncertainty

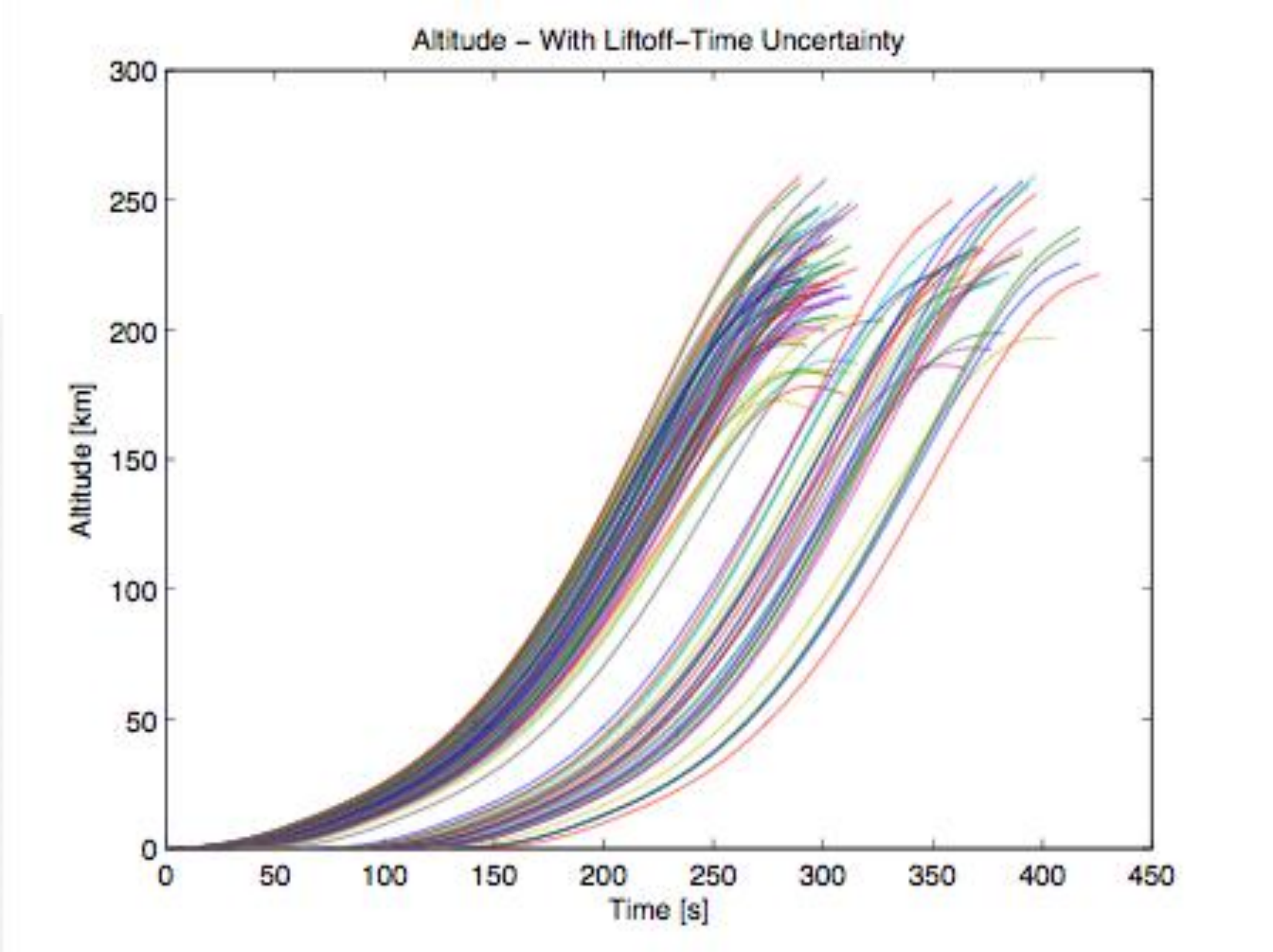
$$C_D = (2 - \cos\alpha)\frac{0.4750M_\infty^2 - 0.7127M_\infty + 0.3049}{M_\infty^2 - 1.914M_\infty + 1.042}$$



Uncertain Lift-off Time

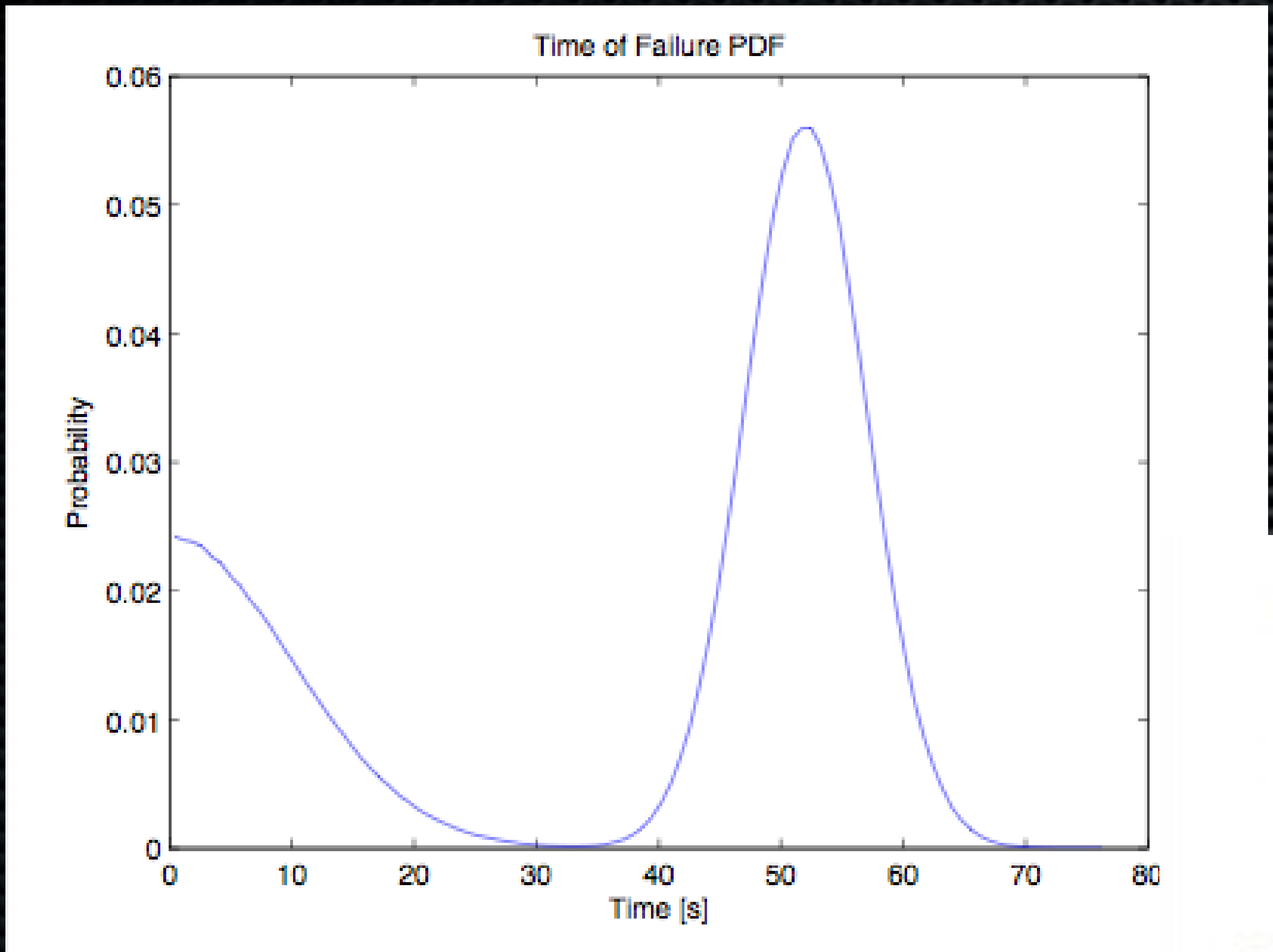


- Rockets do not always launch on time

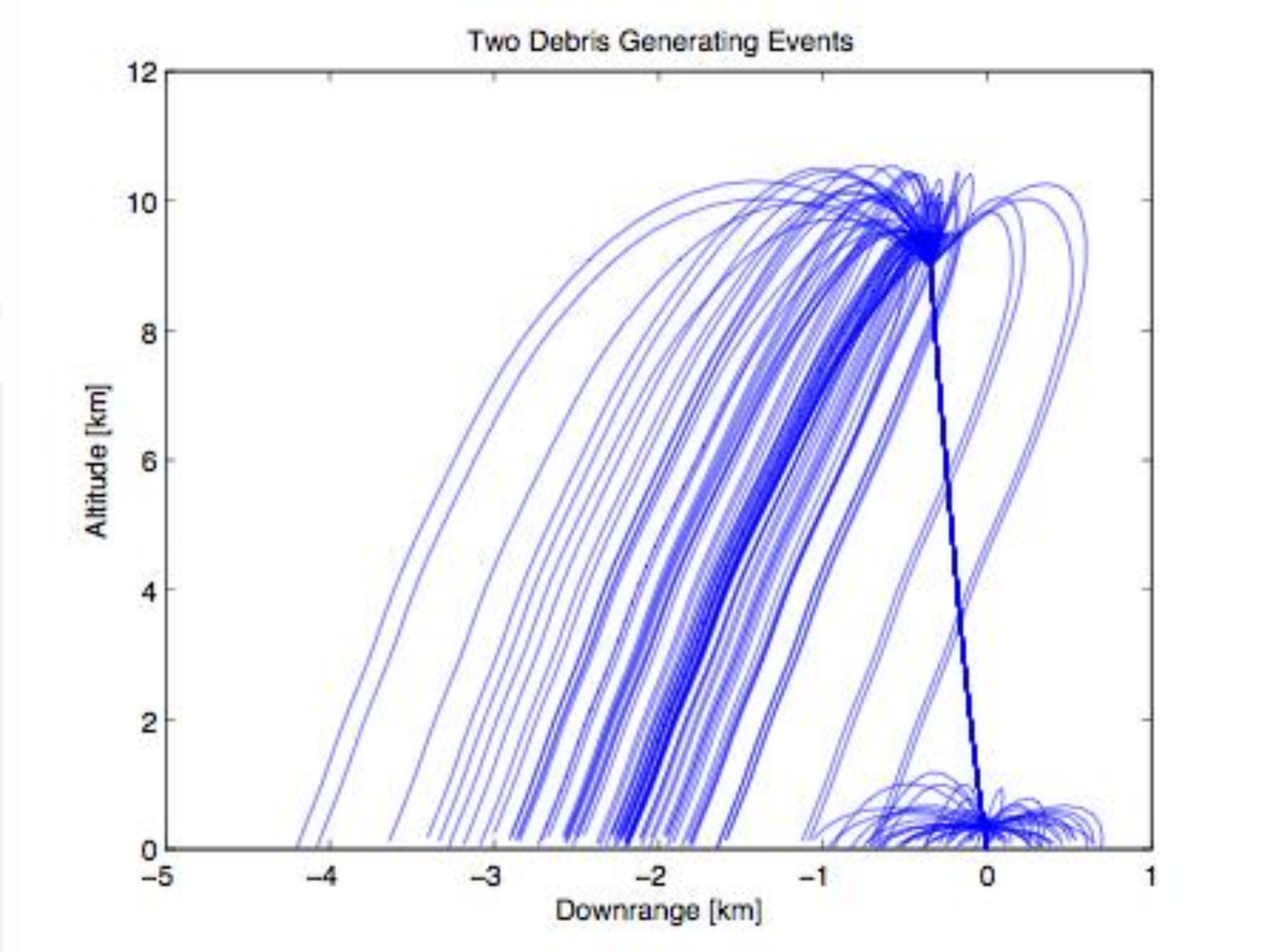


One-sided, multi-modal pdf

Failure Uncertainty



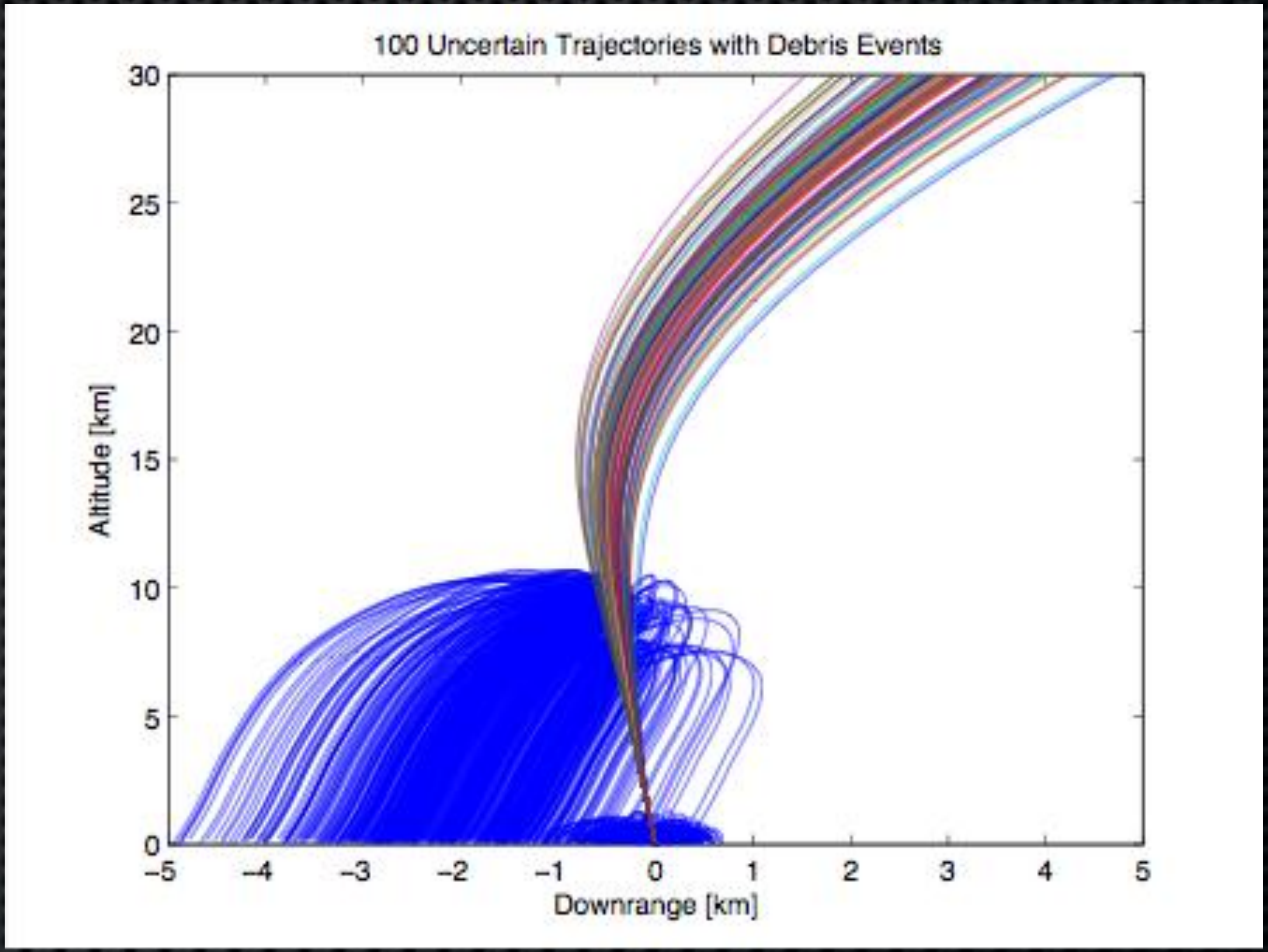
Assume 1% of all launches fail



Failure occurs near pad or at max q

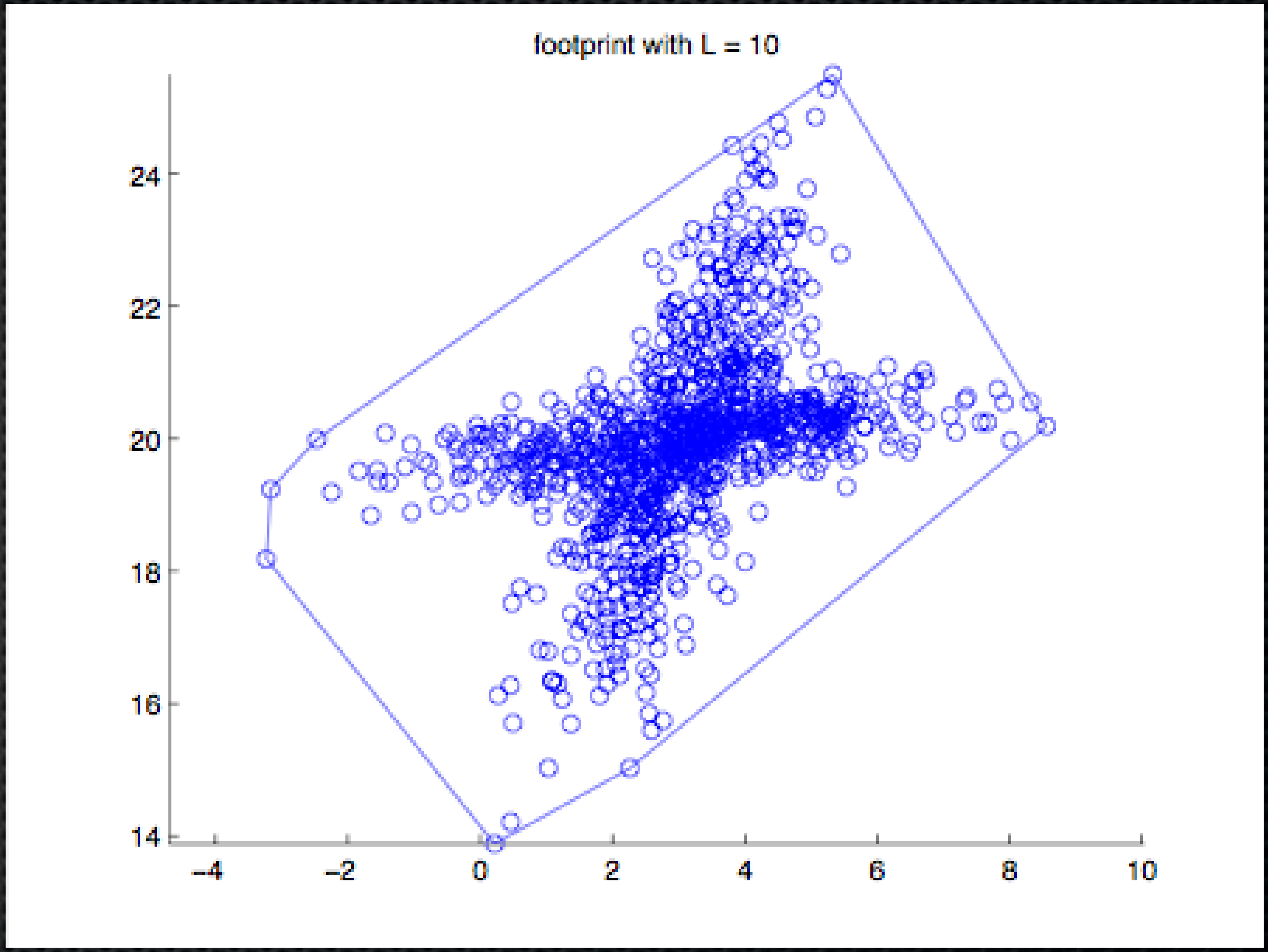
What We've Got So Far

- Software framework that accepts arbitrary:
  - Thrust profiles (TVC, etc)
  - Weather profiles for wind and temperature, with uncertainty parameters for each
  - Failure parameters and distributions
  - Debris model
- Outputs:
  - Collection of uncertain trajectories with debris-generating failure events from a MC simulation



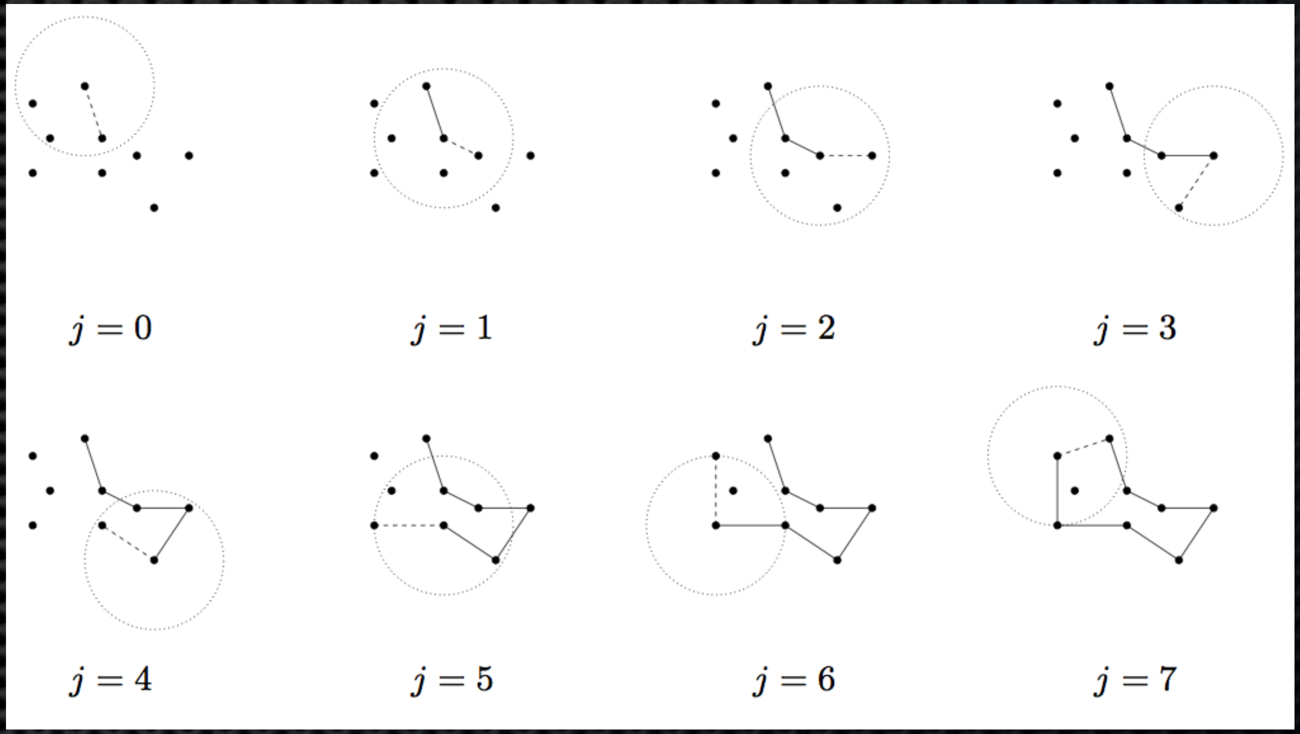
4D Probabilistic Trajectories and Envelopes

- Trajectories as points in space and time
- Risk level of 10^-10, approximated with MC
- How do we turn this set of trajectories into something useful?
- Methods Available
  - Level Sets
  - Delauney Triangulation
  - Convex Hulls
  - Non-convex Footprints



Swinging Arm

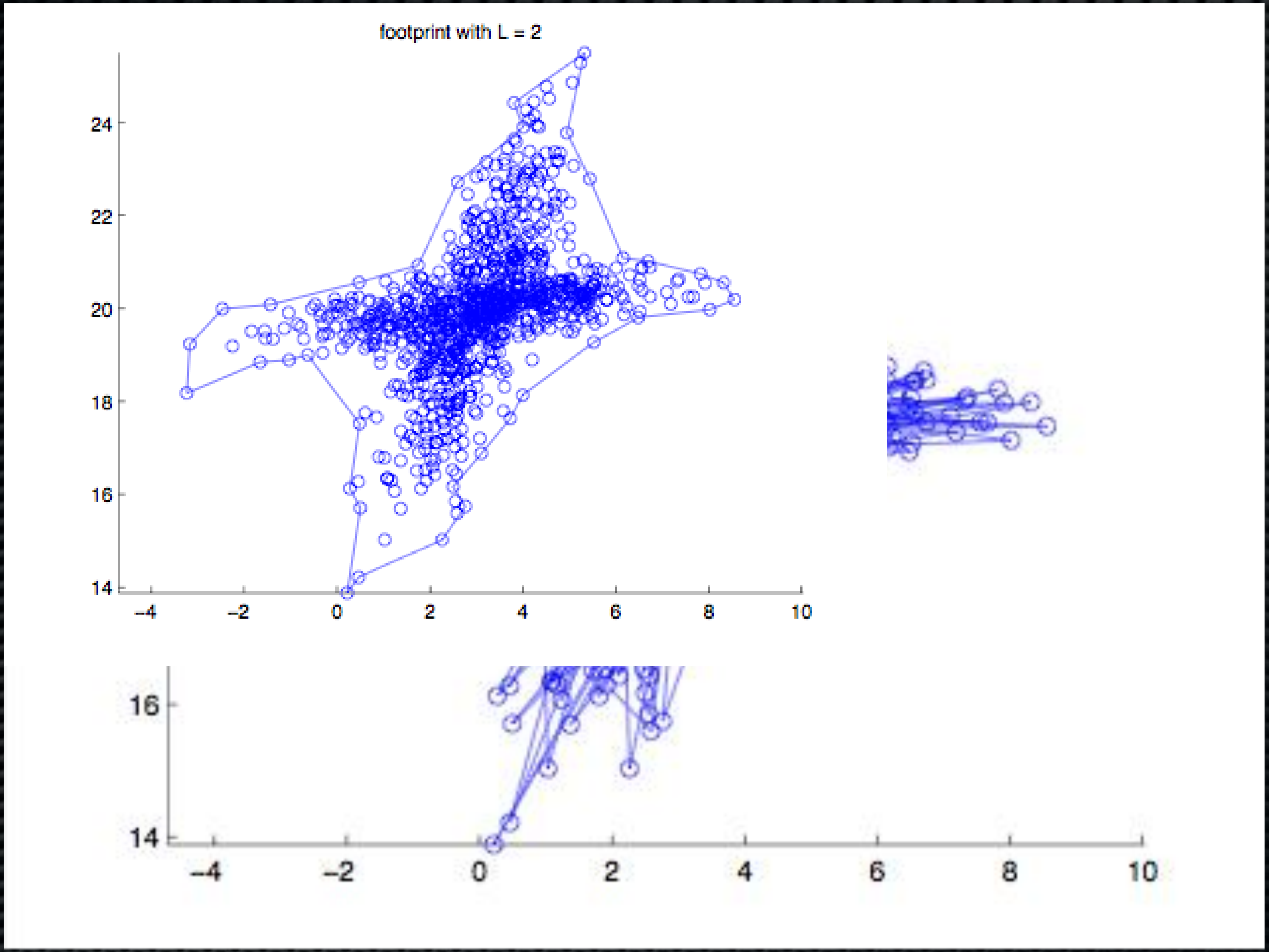
- Generates multiple disconnected "footprints"
  - Non-convex, non-regular polygon
- Creates groupings that visually appear more accurate
- Generalizes up to 3D
- Arm short enough, multiple footprints



Source: Galton, A. "What is the region occupied by a set of points?"

- Cons:
  - Non-regular polygons

Footprint Example



Footprint Example

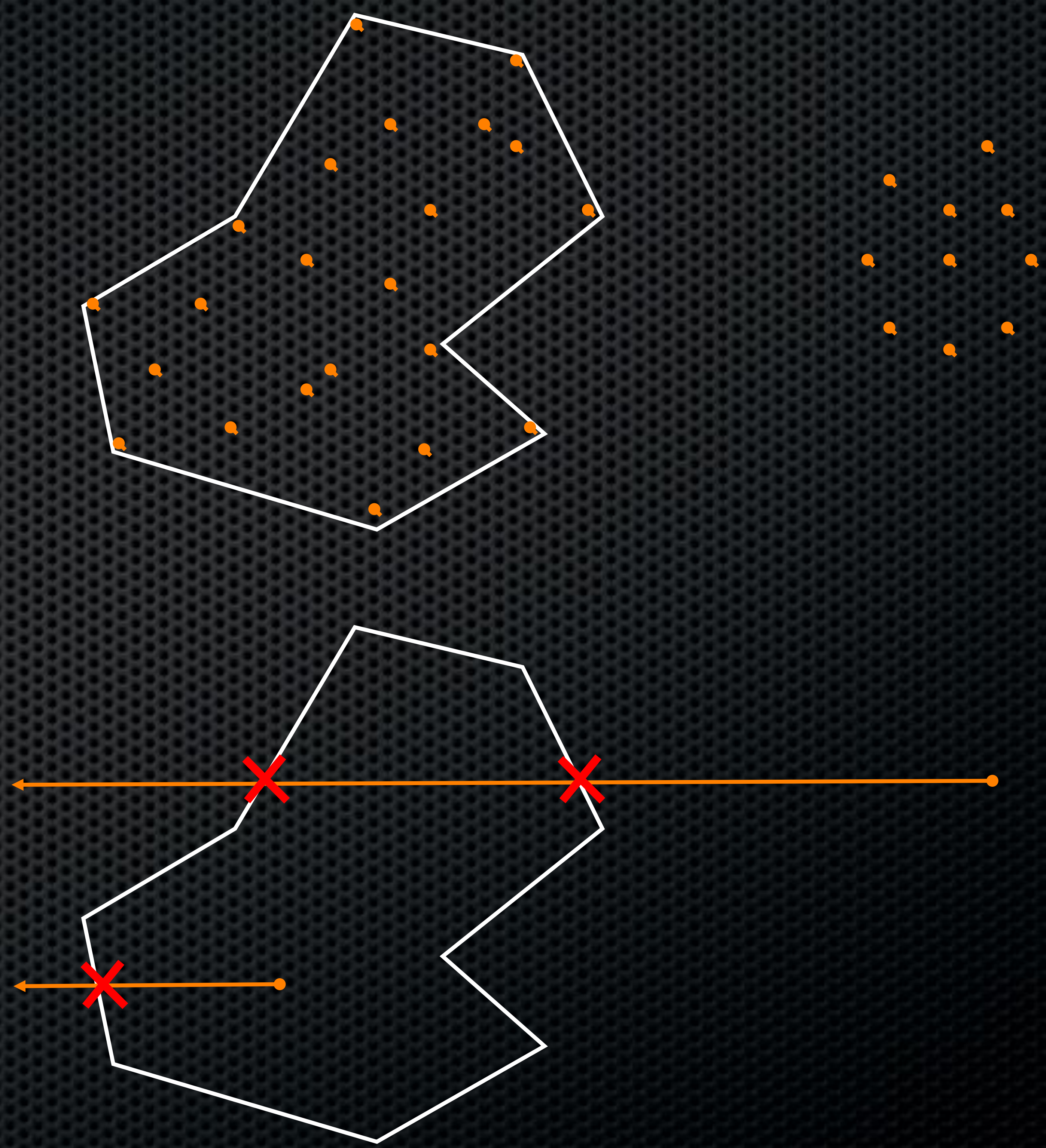
QuickTime™ and a
decompressor
are needed to see this picture.

Making the next footprint

- Arm length short enough, get multiple footprints

- Remove interior and boundary points

- Crossings:
  - Odd is in
  - Even is out



An Early Footprint

QuickTime™ and a
decompressor
are needed to see this picture.

Footprint Through NAS (L=40km)

QuickTime™ and a
decompressor
are needed to see this picture.

Footprint Through NAS (L=4km)

QuickTime™ and a
decompressor
are needed to see this picture.

Footprint Through NAS (L=2km)

QuickTime™ and a
decompressor
are needed to see this picture.

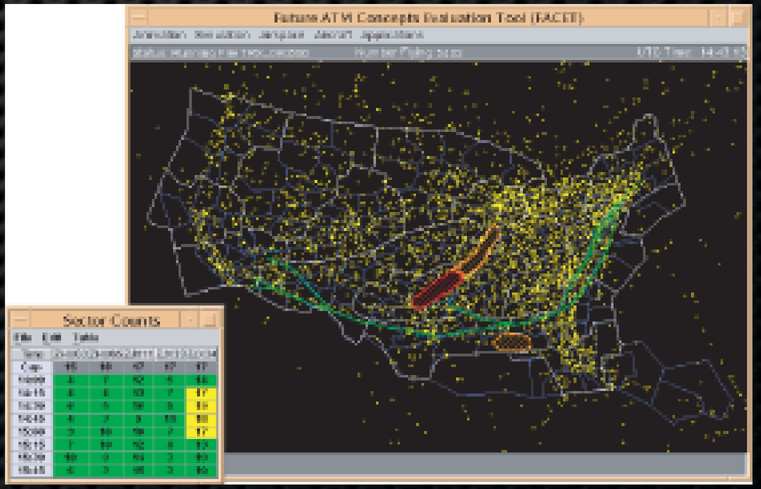
Volume Savings

- Tube: 51,400 km2 sec
  - Conservative. No safety factors.
- Convex: 15,300 km2 sec
  - 30% of the original volume
- Footprint 2km arm: 6,500 km2 sec
  - Only 13% of the original volume!

QuickTime™ and a
decompressor
are needed to see this picture.

Conclusions & Future Work

- Conclusions:
  - Code accepts arbitrary thrust, weather, and failure profiles for Monte Carlo simulation of uncertain trajectories
  - Creates multiple polygonal envelopes around the trajectories (and debris) that represent a no-fly zone
  - Demonstrates the possibility of significant volume (area*sec) savings over conventional tube approach



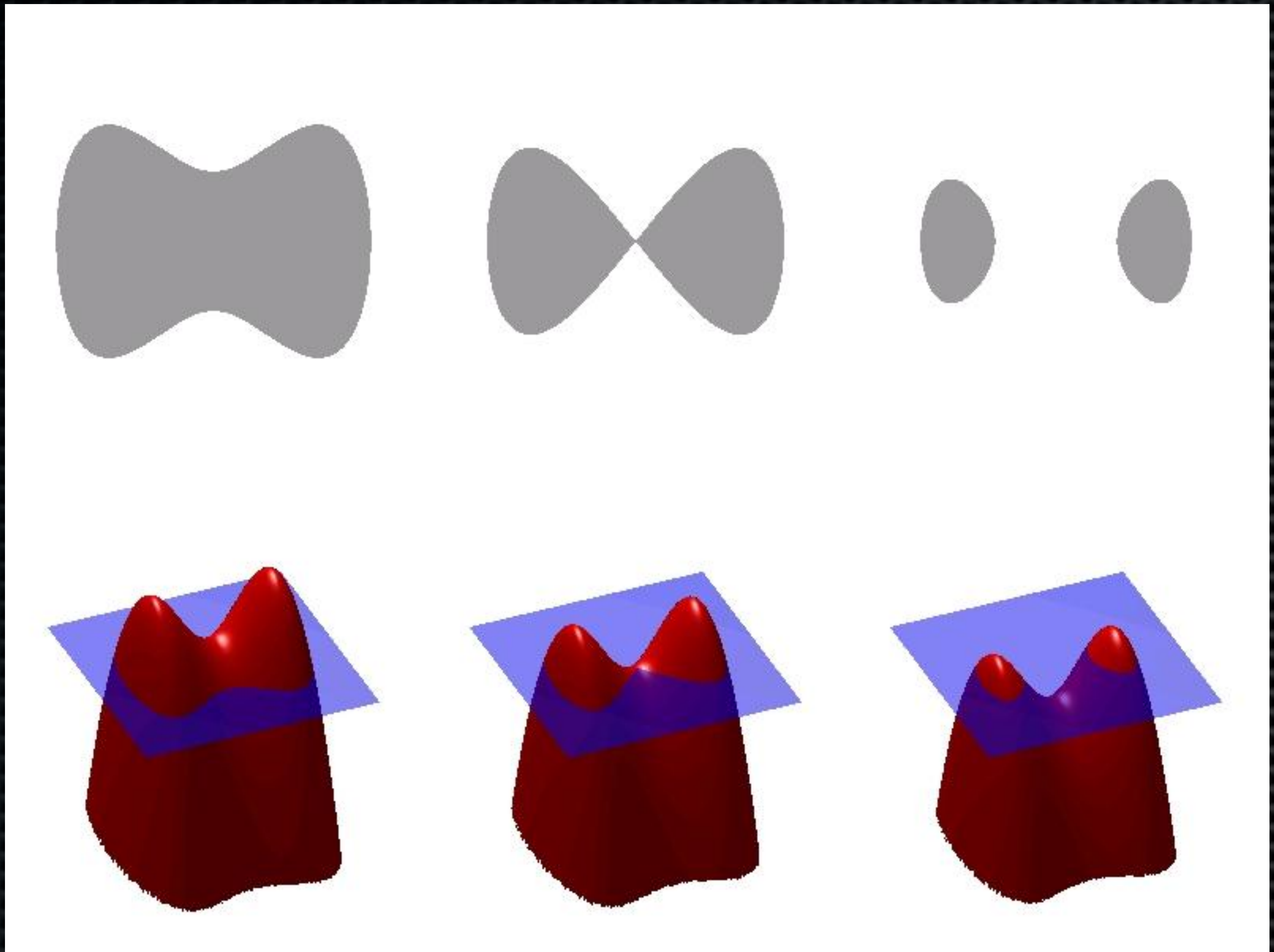
- Future Work:
  - Full 4-D (Swinging Slab)
  - Accurate weather and debris models with uncertainty
  - Active control in rocket during ascent and staging
  - Integration with NASA's FACET tool for scenarios with various launch sites, frequencies + typical day in the NAS

References

- De Berg, M., et al. “Computational Geometry - Algorithms and Applications”, Springer 1998
- Capristan, F. “Aerodynamic Effects in Launch Vehicle Optimal Trajectories”, Stanford 2010
- Osher, S.J., Fedkiw, R.P. “Level Set Methods and Dynamic Implicit Surfaces”, Springer 2002
- Galton, A., Duckham, M. “What is the region occupied by a set of points?” GIScience 2006, LNCS 4197, pp. 81-98, 2006
- Goldman, R. "Intersection of Two Lines in Three-Space." In Graphics Gems I (Ed. A. S. Glassner). San Diego: Academic Press, p. 304, 1990.
- Colonno, M. R., S. Reddy, and J. J. Alonso. "Multi-Fidelity Trajectory Optimization with Response Surface-Based Aerodynamic Prediction." (2008)
- Stengel, Robert. "Launch Vehicle Design: Trajectories and Aerodynamics." Launch Vehicle Design Class Notes. N.p., n.d. Web. 26 May 2010. <http://www.princeton.edu/~stengel/MAE342Lecture3.pdf>.

Backup Slides

Level Sets



Source: http://en.wikipedia.org/wiki/Level_set_method

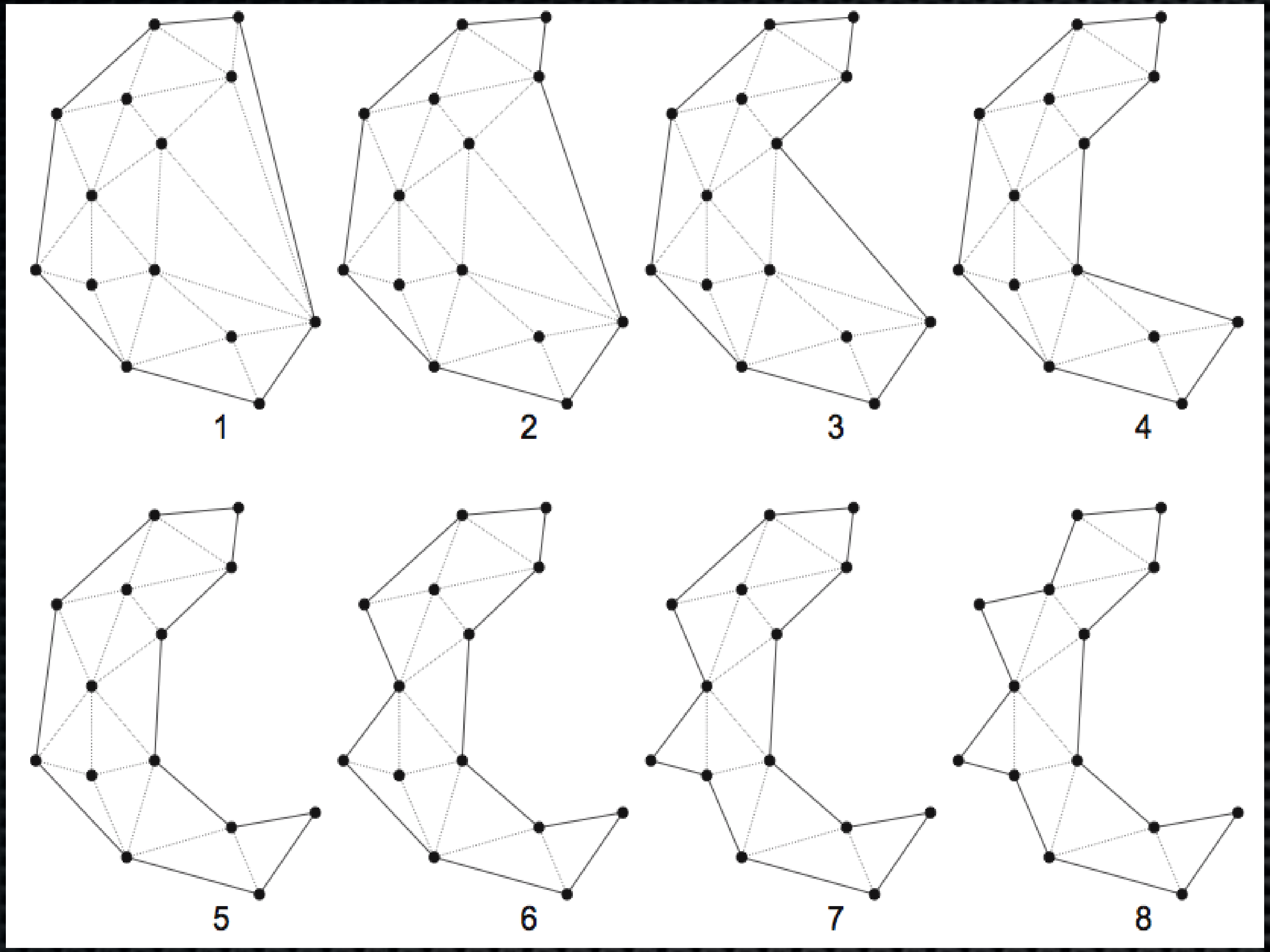
- Useful for visualizing dynamic interfaces
- N-Dimensional surface is slice of an (N+1)D function
- Easily handles pinching and merging interfaces
- Set operations are easy

Level Set Example

- Hard to create the distance function
- Finding the area enclosed is not straightforward
- Allows holes within the boundary
- Slow

QuickTime™ and a
decompressor
are needed to see this picture.

Delauney Triangulation

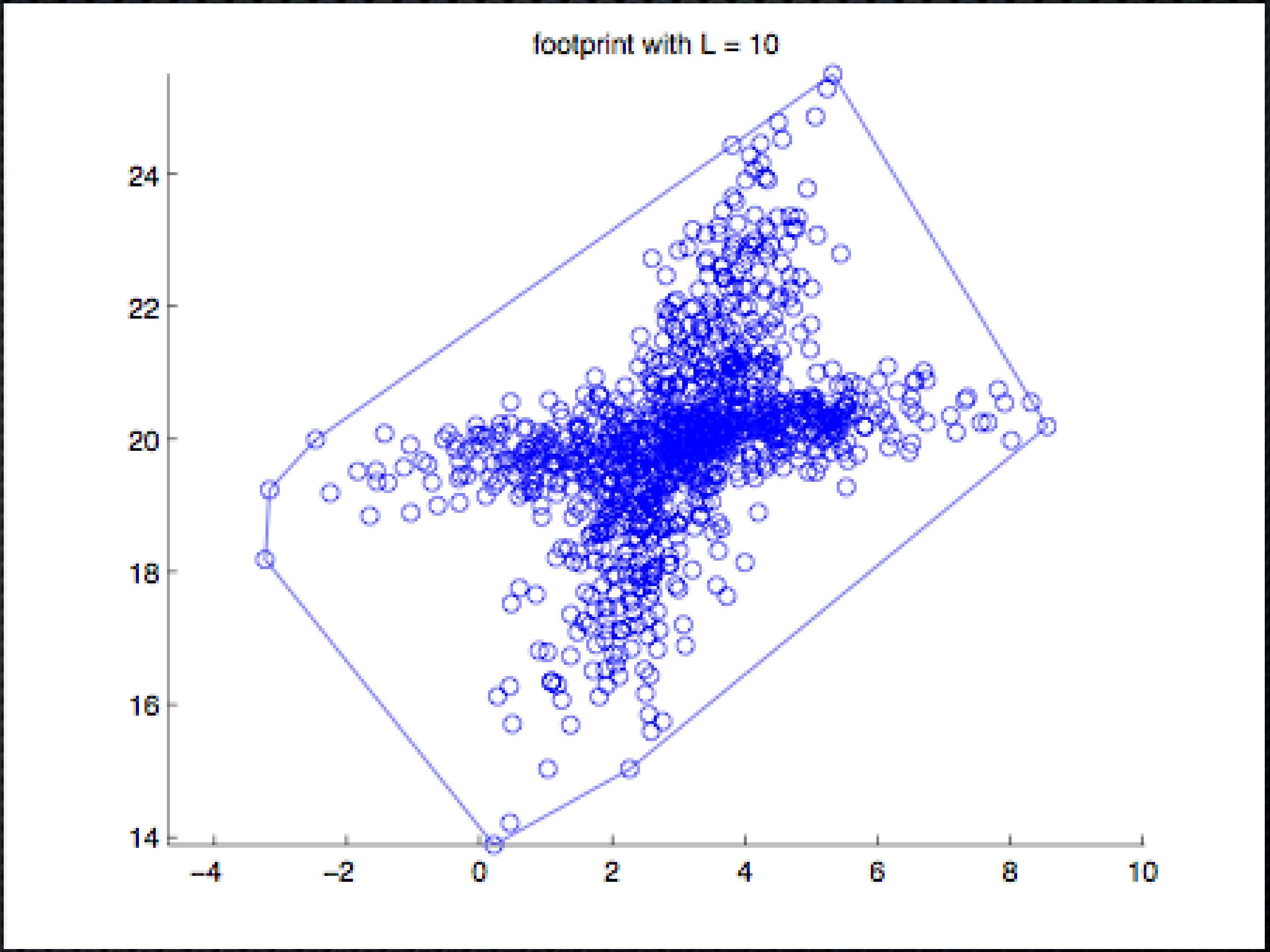


Source: Galton, A. “What is the region occupied by a set of points?”

- Overview
- Connect all dots with series of triangles
- Remove boundary edges
    - Generates single connected regular polygon
- Cons:
- Want to eliminate most points! Worth it?
- Creates a single shape

Convex Hulls

- Easy to generate
- Wastes a lot of space
- Only get one shape
- Can get these with footprint methods



Swinging Arm Algorithm

- Order all points from top to bottom, right to left.
- Set all points as ‘available’ and pick an arm length
  - Store top-right available point in footprint and set it as current point:
    - Swing the arm clockwise from current point until it hits another point
    - Store this point as being in the footprint and set it as the new current point
    - Repeat until current point == starting point
  - Set all points that form or are interior to the footprint as ‘unavailable’
  - Repeat until all points are unavailable
- xxxx