The practitioners guide to SDDP

Experiences from `SDDP.jl`

Oscar Dowson

Department of Engineering Science
University of Auckland

June, 2018

Outline

`SDDP.jl`

Scary things with SDDP

- Degeneracy
- Numerical Issues

A crisis of reproducibility

Outline

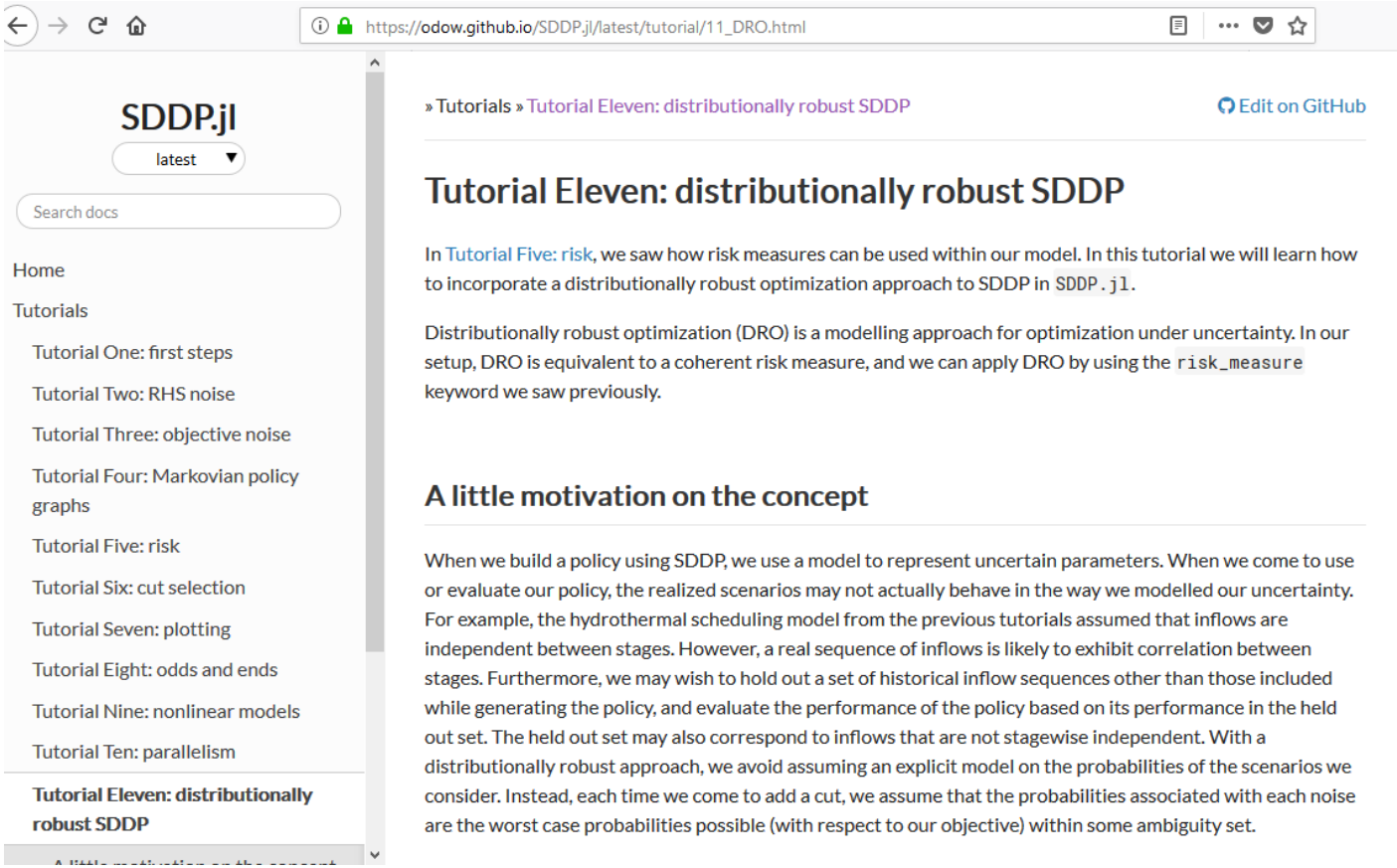
SDDP.jl

What is it?

- A generic SDDP library in the Julia language
- Built upon JuMP $\implies$ nice syntax
- Similar performance to C++ implementation
- Some cool features:
  1. User-extensible risk measures
  2. User-extensible cut selection heuristics
- Google "SDDP.jl github"

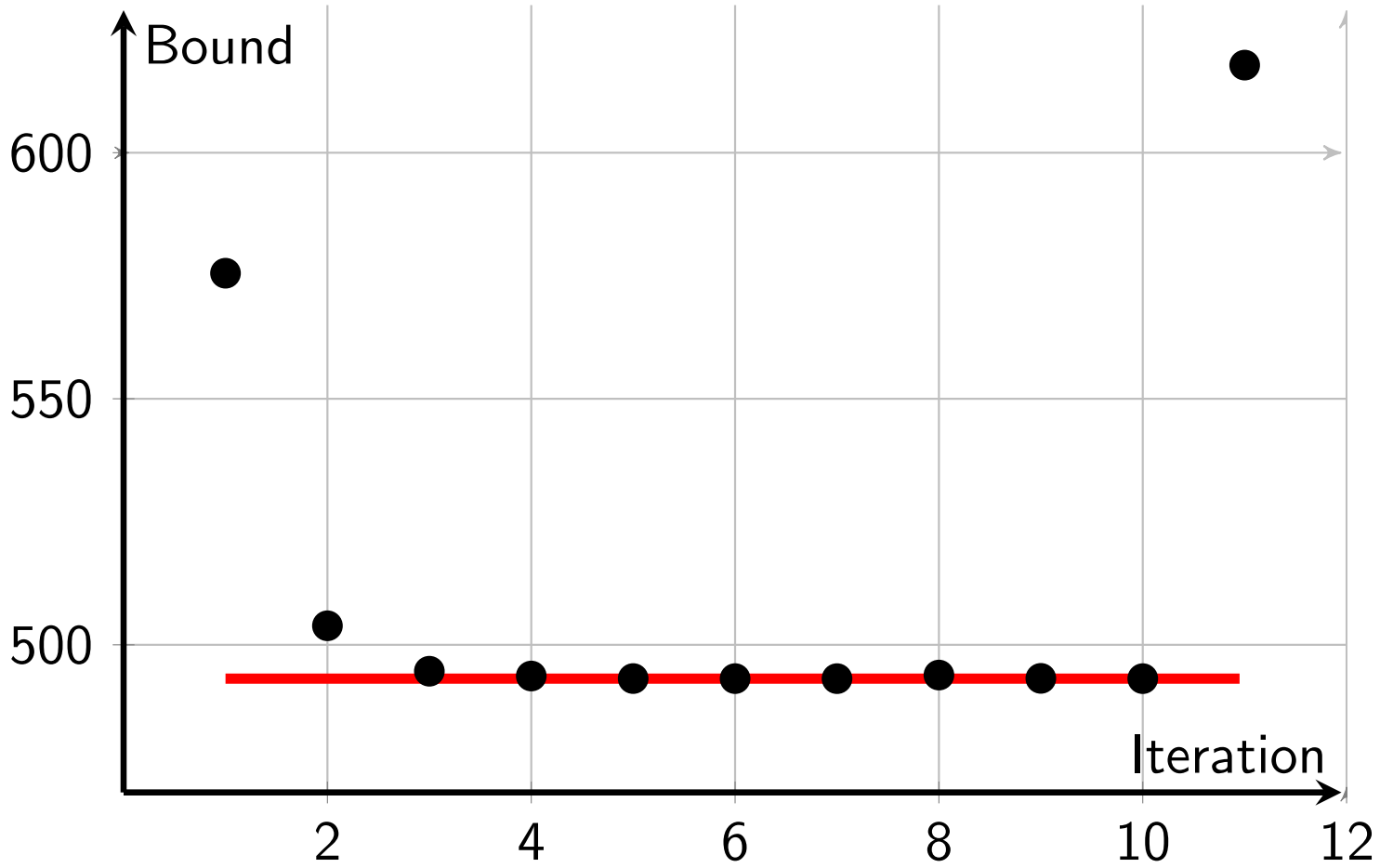
SDDP.jl

Success story: the paper of Philpott, de Matos, Kapelevich (2018) used `SDDP.jl` to implement a DRO version of SDDP in < 50 lines of code.

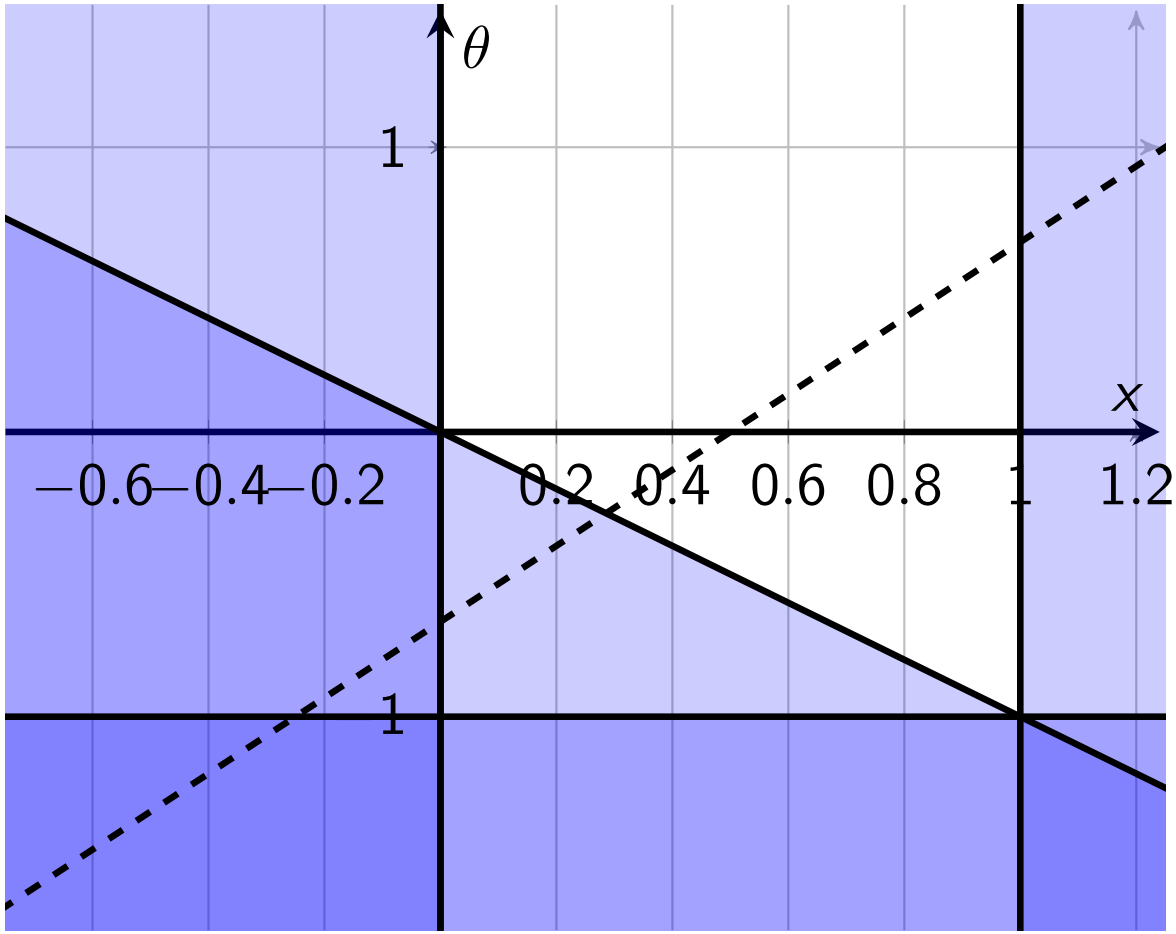


Outline

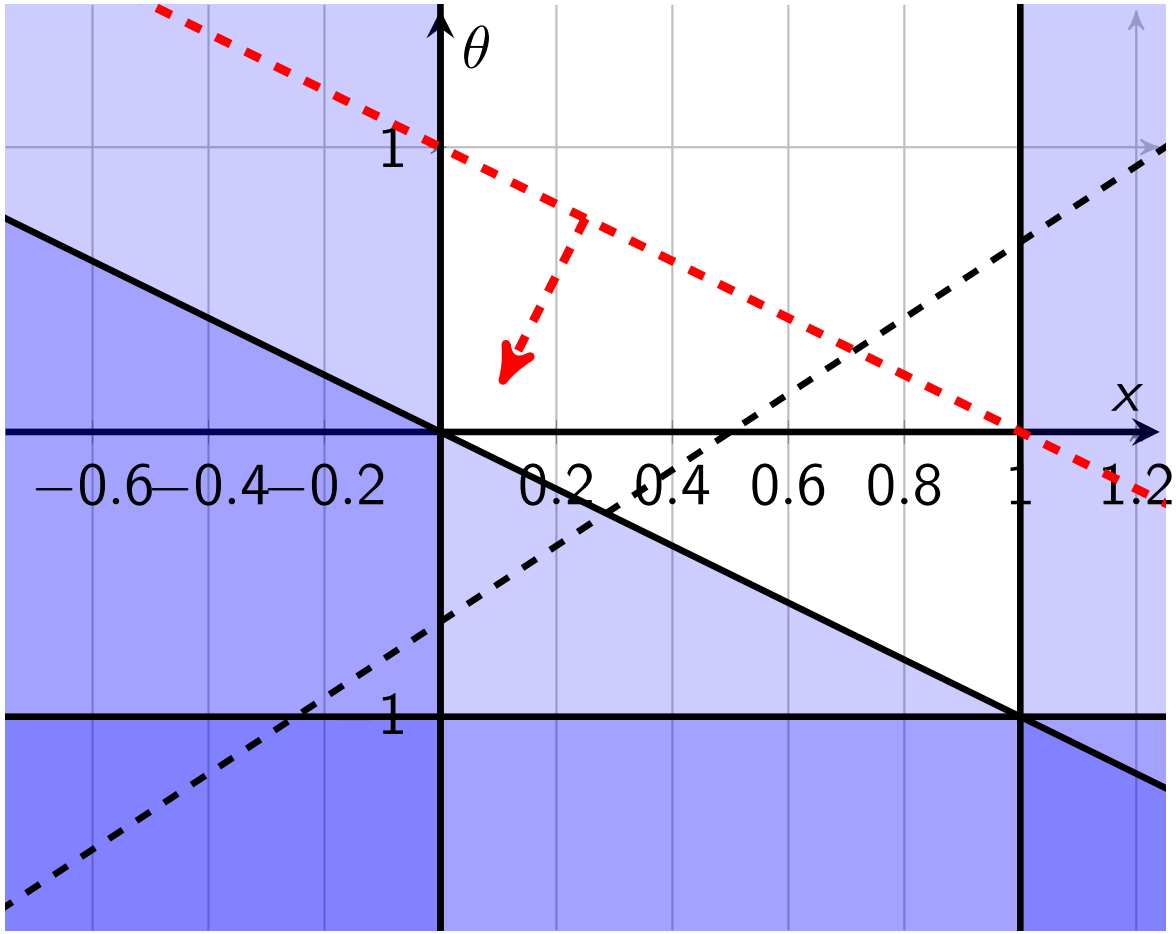
Degeneracy



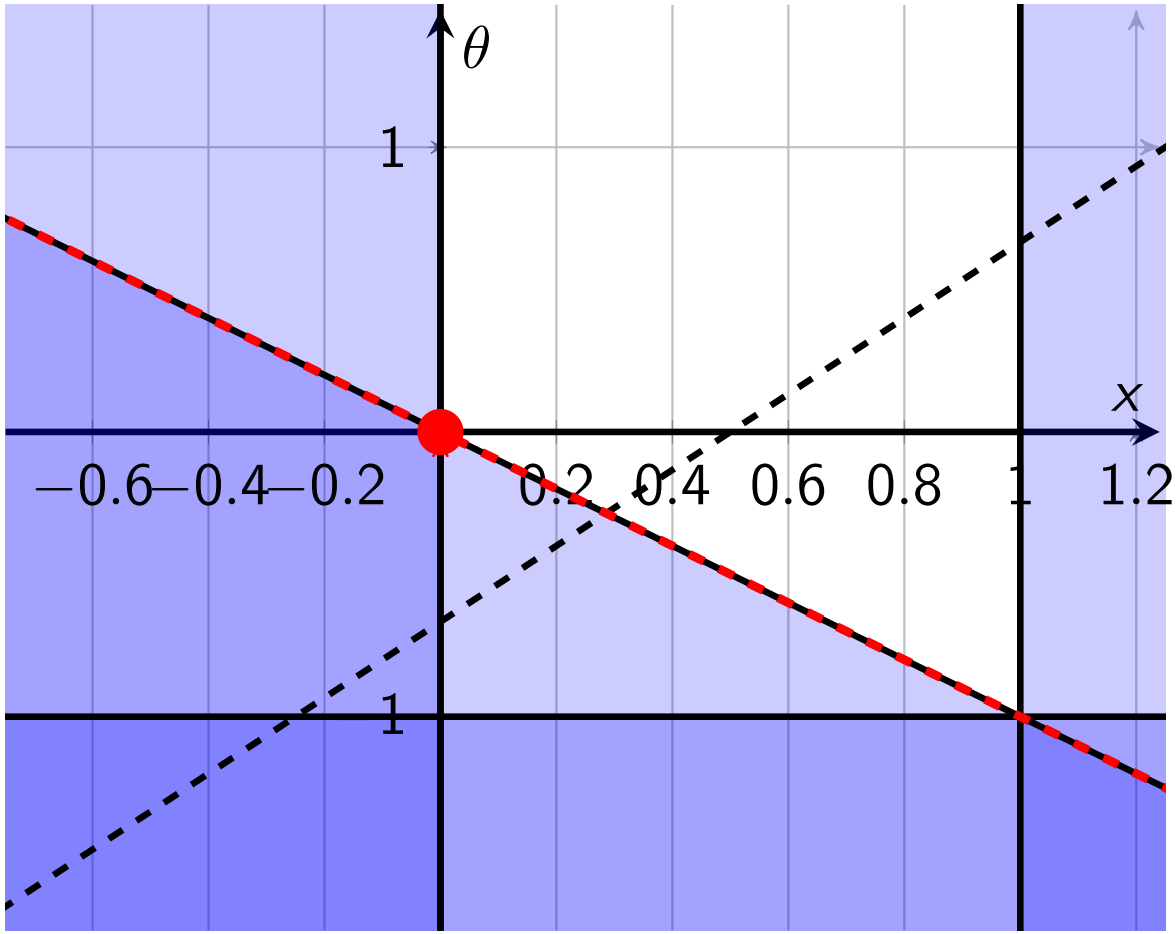
Degeneracy



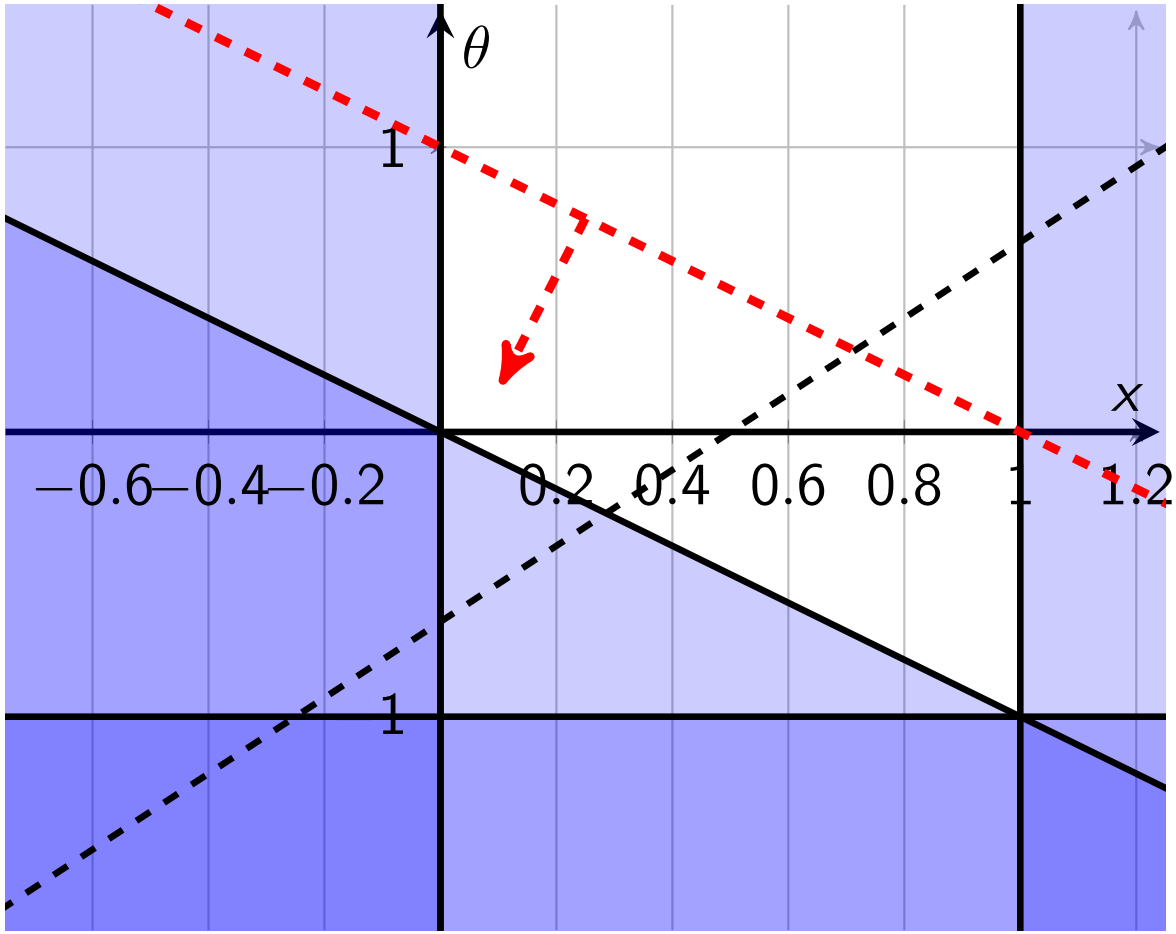
Degeneracy



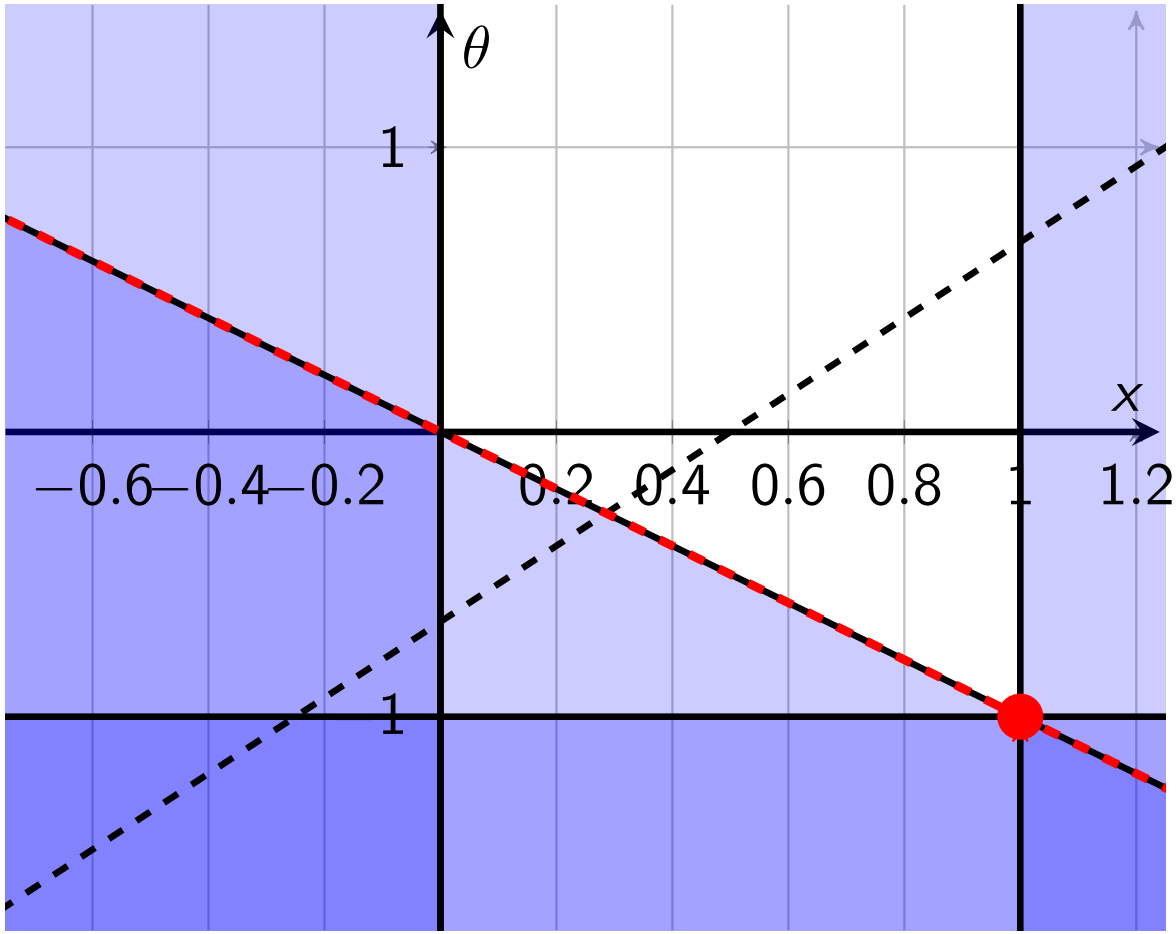
Degeneracy



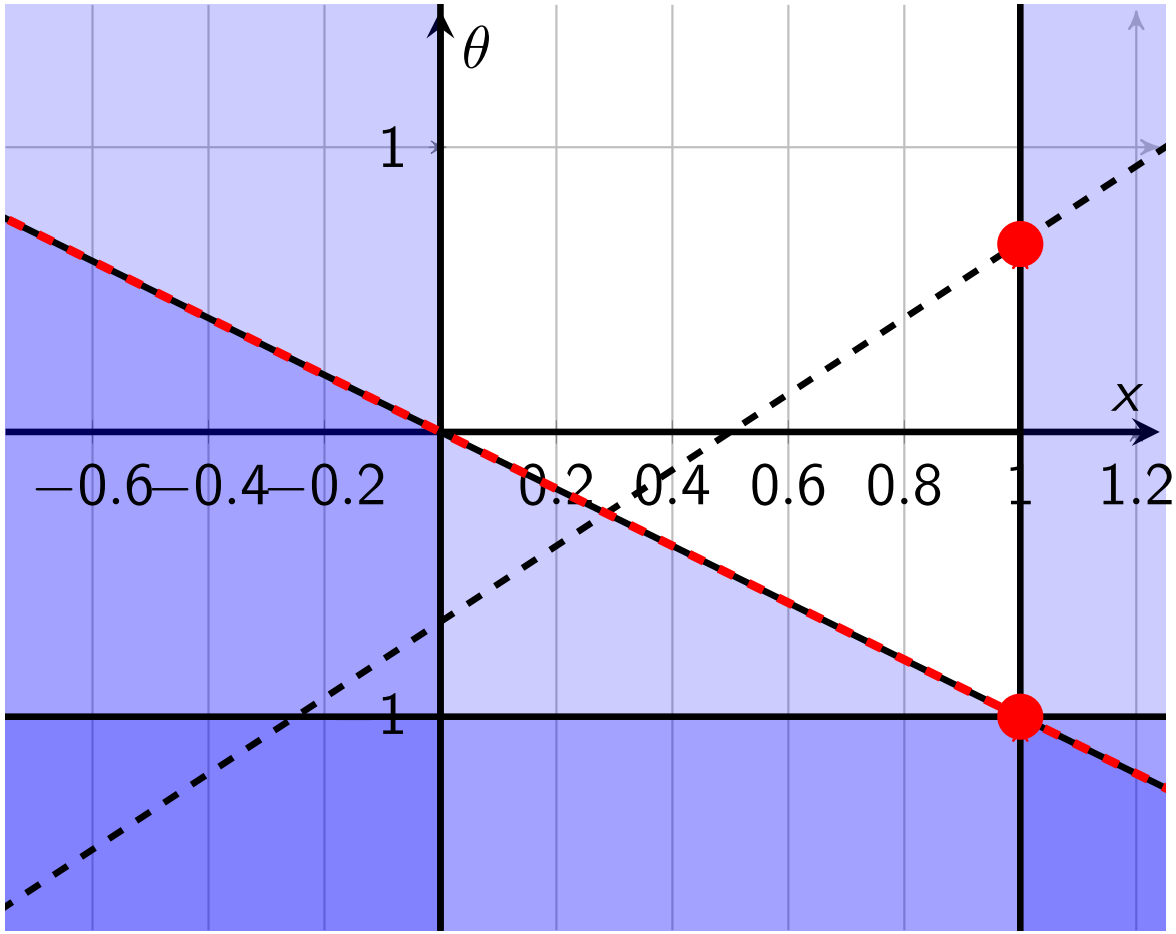
Degeneracy



Degeneracy



Degeneracy



Degeneracy

Implication for SDDP

When we simulate an "optimal" policy, we may obtain a sequence of sub-optimal controls, even in a converged, deterministic model.

Numerical Issues

1. There might be a bug in `SDDP.jl`
2. There might be a bug in your code
3. There might be bugs in JuMP or Julia
4. We have found bugs in Gurobi. Wrong solutions to simple LP's
5. Clp will willingly provide numerically incorrect solutions

Numerical Issues

Implication for SDDP

How do we know a solution is correct? It is hard to validate the solution of a multistage stochastic optimisation problem.

Outline

A crisis of reproducibility

SDDP is really cool, but you hear the following all the time:

1. I wrote a (private) implementation in XXX language
2. It is hard-coded to solve my favourite (energy-related) problem XXX
3. The standard algorithm is too slow, so I made it XXX times faster by implementing MyNewPublishedTechnique^{TM}
4. Everything worked beautifully and I got a nice answer.

A crisis of reproducibility

1. We don't share code
   (so how do I know your implementation is correct?)
2. We can't share models
   (so I can't test my method on your problem)
3. We use the same words with different meanings
   (so I can't easily understand your paper and re-implement)

A crisis of reproducibility

1. We don't share code
   (so how do I know your implementation is correct?)
2. We can't share models
   (so I can't test my method on your problem)
3. We use the same words with different meanings
   (so I can't easily understand your paper and re-implement)

How do we know if MyNewPublishedTechnique^{TM} is an improvement?

A proposal

- Let's pick a subset of problems, large enough to be useful, small enough to standardise
- Agree on a common notation, terminology, and model formulation
- Develop a file-format
- Develop a set of open-source solvers that can read the format
- Develop a set of test problems

A proposal

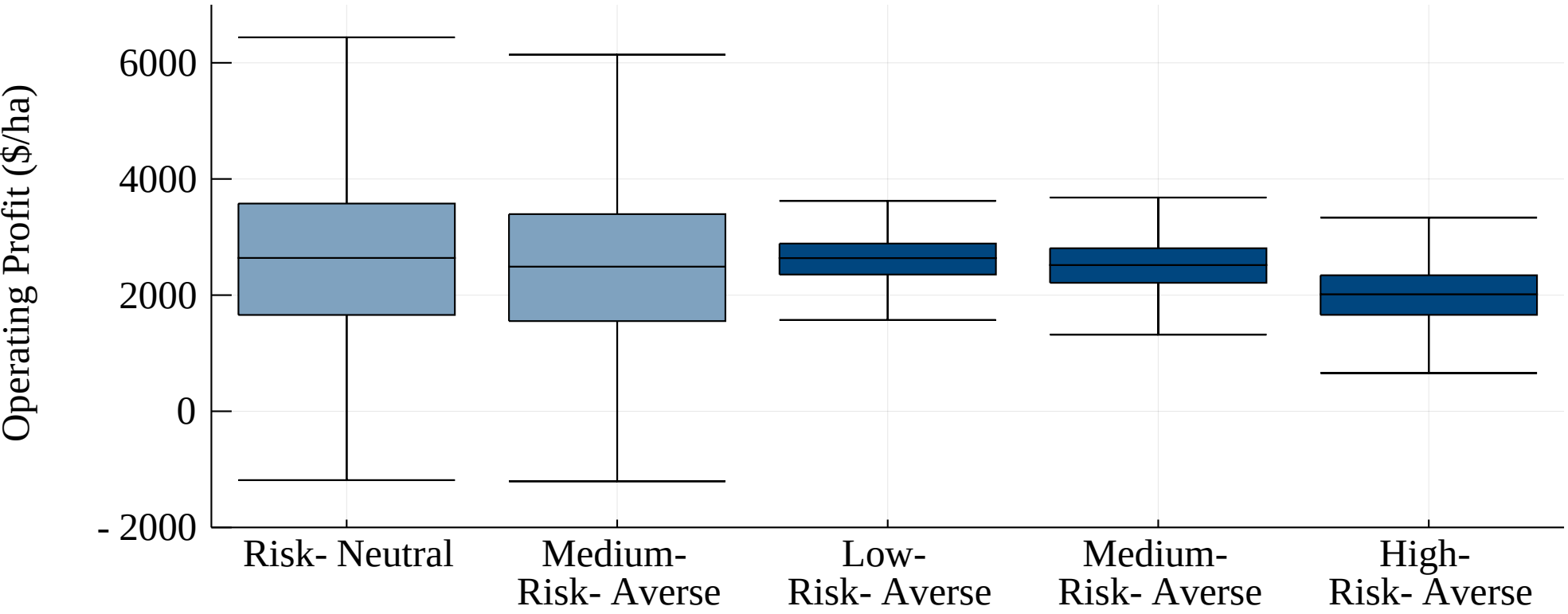
THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

- Let's pick a subset of problems, large enough to be useful, small enough to standardise
- Agree on a common notation, terminology, and model formulation
- Develop a file-format
- Develop a set of open-source solvers that can read the format
- Develop a set of test problems
- If MyNewPublishedTechnique^{TM} can solve the test problems faster it is an improvement.

A proposal

Questions?

Nested Risk Measures



Nested Risk Measures

Implication for SDDP

Nested risk-measures don't match our intuition, and they don't produce consistent results.