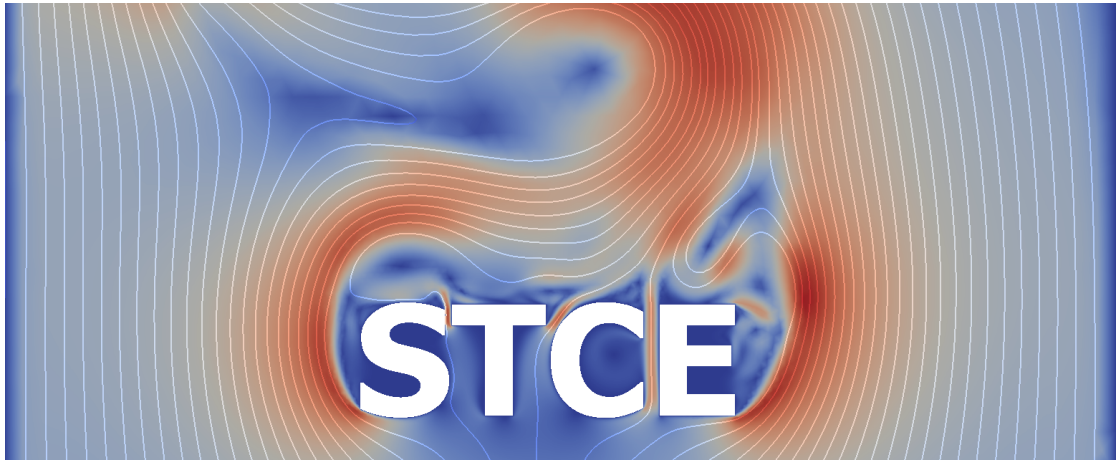


Significance Analysis for Numerical Models

... within the FET-open project SCoRPiO

U. Naumann J. Riehme
Software and Tools for Computational Engineering Science
RWTH Aachen University

1st Workshop On Approximate Computing (WAPCO 2015), Amsterdam

Outline

SCoRPiO

Significance Analysis for Numerical Models

- Objective
- Algorithmic framework
- Significance Analysis in a nutshell
- Applications

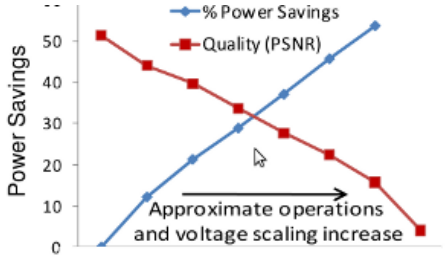
Outlook

SCoRPiO— Significance-Based Computing for Reliability and Power Optimization

SCoRPiO

Significance-Based Computing for
Reliability and Power Optimization

SEVENTH FRAMEWORK PROGRAMME



Power savings through controlled quality degradation, even in the presence of hardware faults

CERTH / IRETETH

RWTH AACHEN UNIVERSITY

Inria INVENTORS FOR THE DIGITAL WORLD

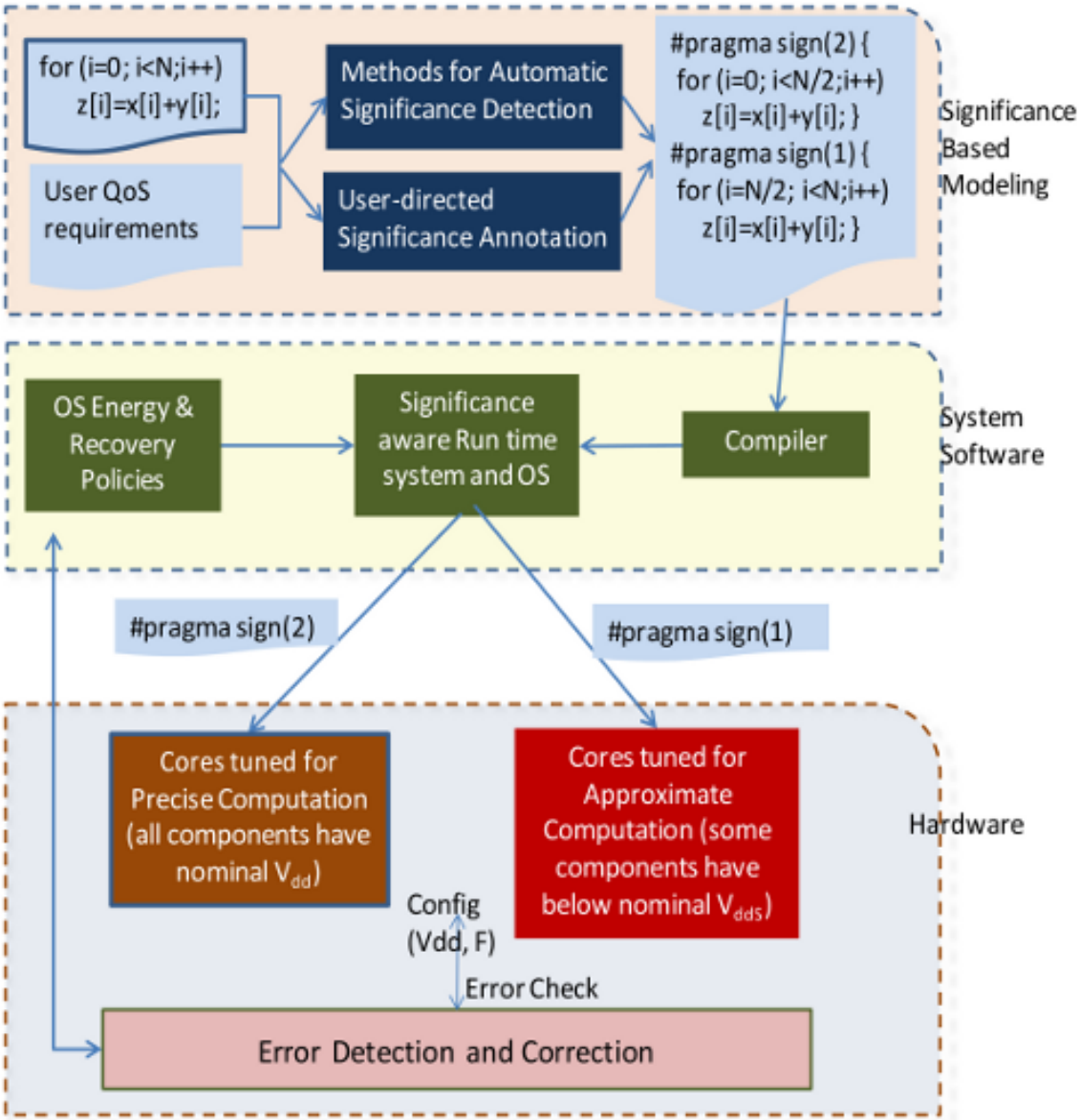
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Queen's University Belfast

imec

A multi-national, multi-disciplinary group of researchers

SCoRPiO
Project Structure



Significance Analysis of Numerical Models

Significance Analysis for Numerical Models
Objective

Given:

- Implementation of an algorithm (code) in C or C++
- Ranges of input values
- Rate of acceptable variability in output ranges (very application specific)

Objective:

Identify parts of the given code, that are **insignificant** wrt the user given allowed variability rate / bound / . . . for the output ranges.

A insignificant computation might be replaced by a cheaper computation

- using unreliable hardware
- using lower precision on slower, less power consuming hardware
- replacing the computations by a fixed (constant) representative
- reusing previously computed results for subsequent computations

(One) Motivation:

Iterative algorithms tend to find a good solution in early stages.

Significance Analysis for Numerical Models
Algorithmic framework

Key ingredients:

- **Interval arithmetic**
  - Computes guaranteed enclosures of true function values.
  - Value interval of a program variable describes the combined influence of input ranges on the variable (forward analysis).
  - Issues: overestimation (wrapping effect), relational operators.
- **Adjoint mode Algorithmic Differentiation**
  - An adjoint model can compute the individual influence of all inputs on a single output in one run (reverse analysis).
  - Issue: Differentiability is concept for continuous functions.

Technical setup:

- Exploit overloading of operators/intrinsic functions in C++
  - `dco/scorpio` (a specialization of `dco/c++` with `filib++`).
  - Run the executable with significance functionality.
- Nevertheless this a **compile time analysis** for the simulator in SCoRPiO
  - **Output** of significance analysis is **input** for the compiler of the simulator

Significance Analysis in a nutshell (I)

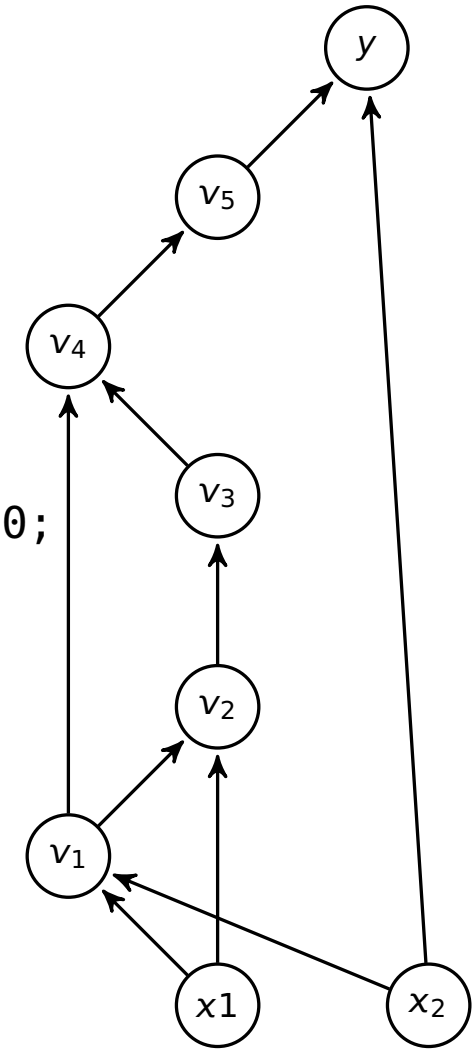
Input data:

$x1, x2 \in [1, 36000]$
$\epsilon = 2.5$

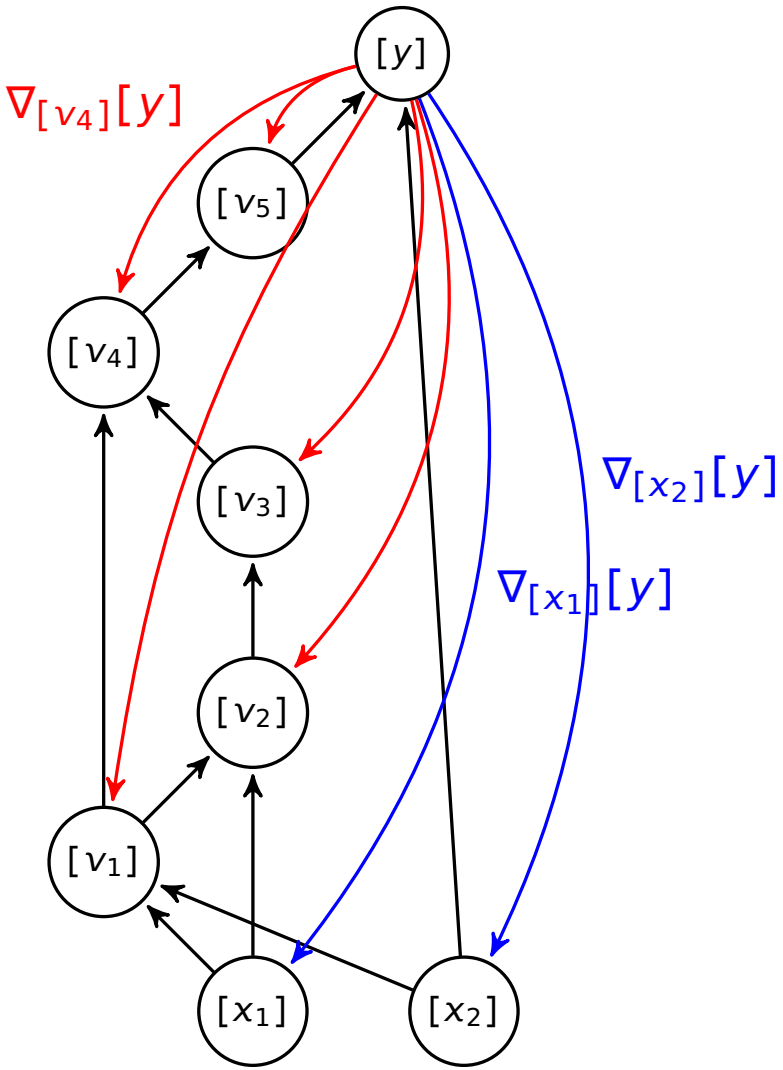
Original code:

```
v1 = x1 * x2;
v2 = sin(x1);
v3 = exp(v2);
v4 = v3 * v1;
v5 = log( v4 );
y  = v2/10 + x2/100;
```

Computational graph



dco/scorpio



Significance Analysis in a nutshell (II)

Input data:

$x1, x2 \in [1, 36000]$
$\epsilon = 2.5$

Original code:

```
v1 = x1 * x2;
v2 = sin(x1);
v3 = exp(v2);
v4 = v3 * v1;
v5 = log( v4 );
y  = v2/10 + x2/100;
```

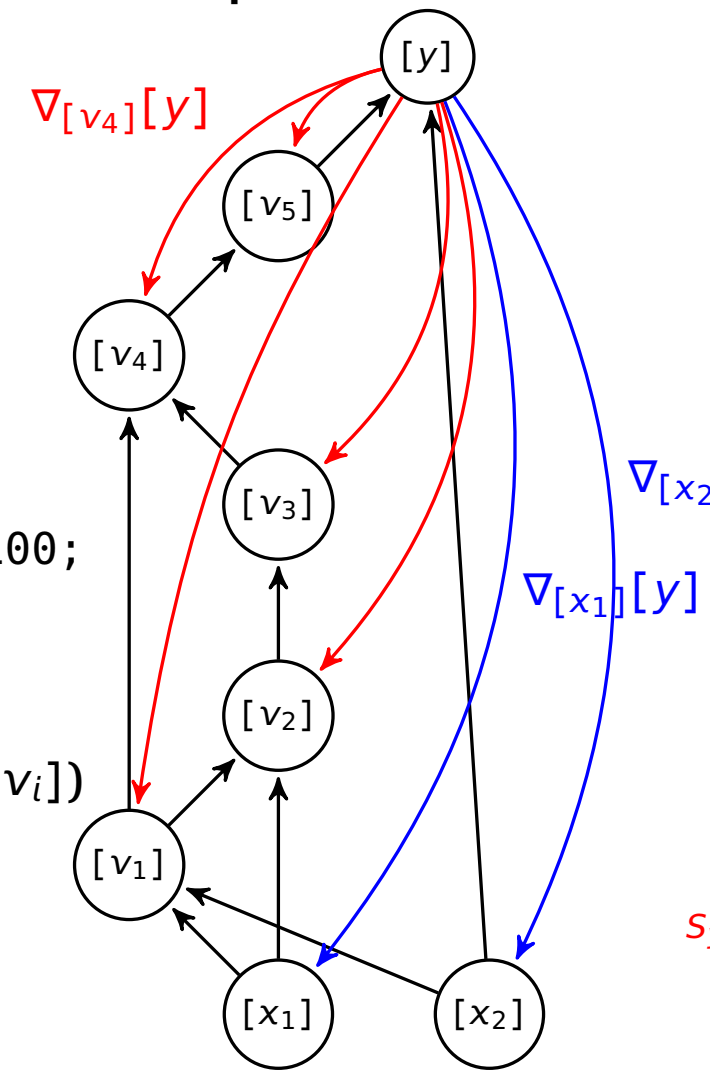
Significance:

$$S_y(v_i) = w([v_i] \cdot \nabla_{[y]}[v_i])$$

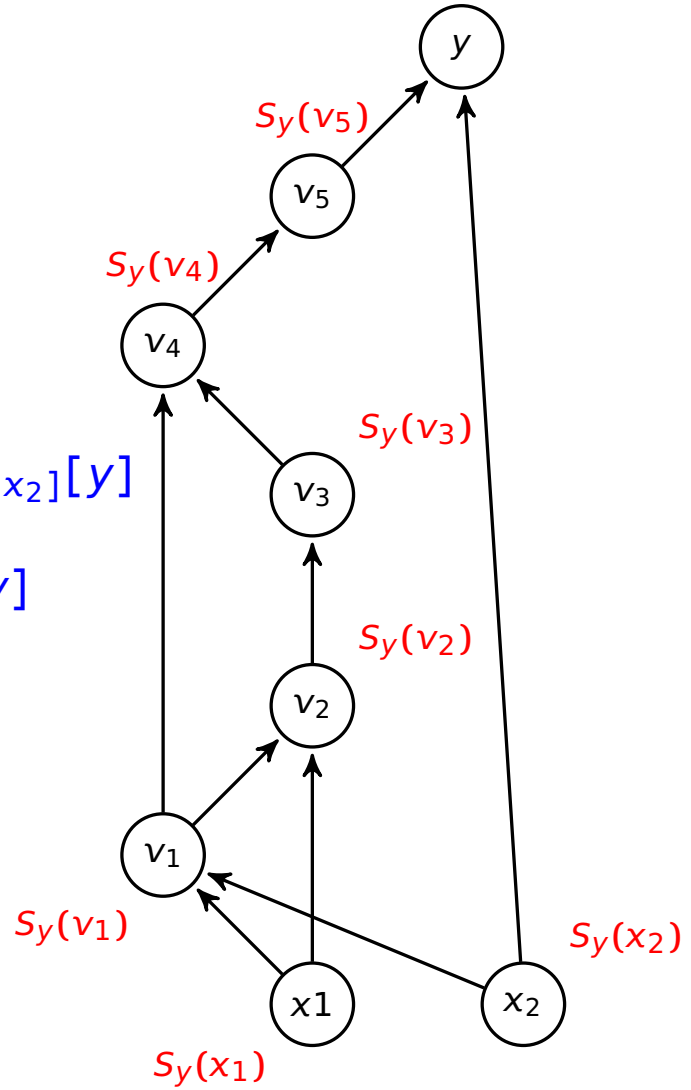
Insignificance test

$$S_y(v_i) \leq \epsilon$$

dco/scorpio



Significance



Significance Analysis in a nutshell (III)

Input data:

$x1, x2 \in [1, 36000]$
$\epsilon = 2.5$

Original code:

```
v1 = x1 * x2;
v2 = sin(x1);
v3 = exp(v2);
v4 = v3 * v1;
v5 = log( v4 );
y  = v2/10 + x2/100;
```

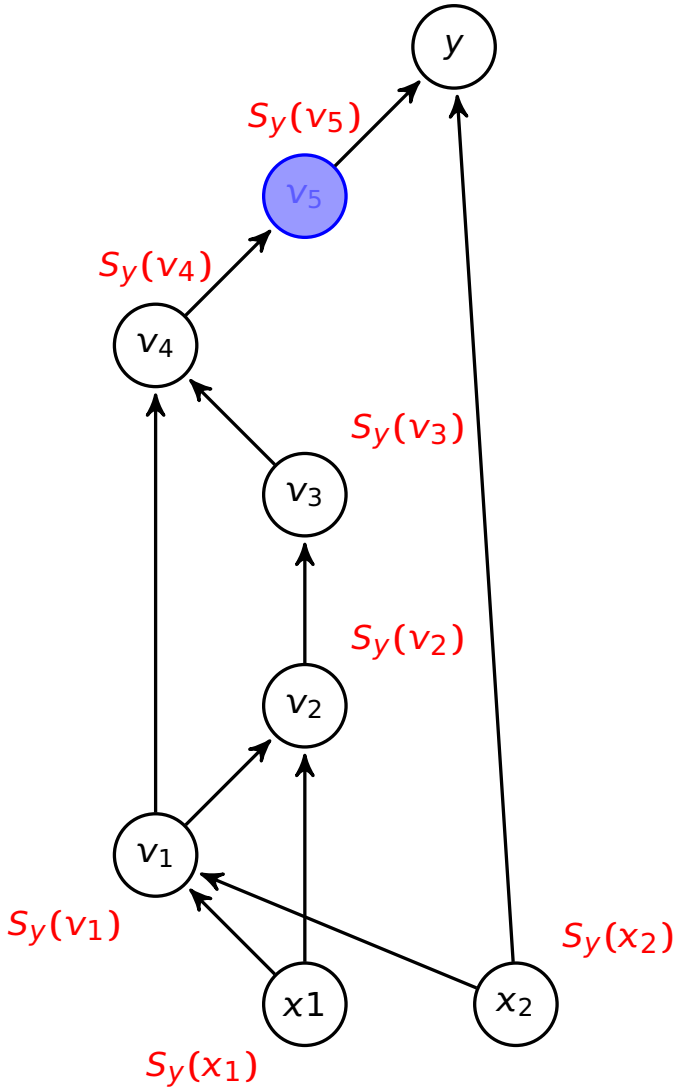
Significance:

$$S_y(v_i) = w([v_i] \cdot \nabla_{[y]}[v_i])$$

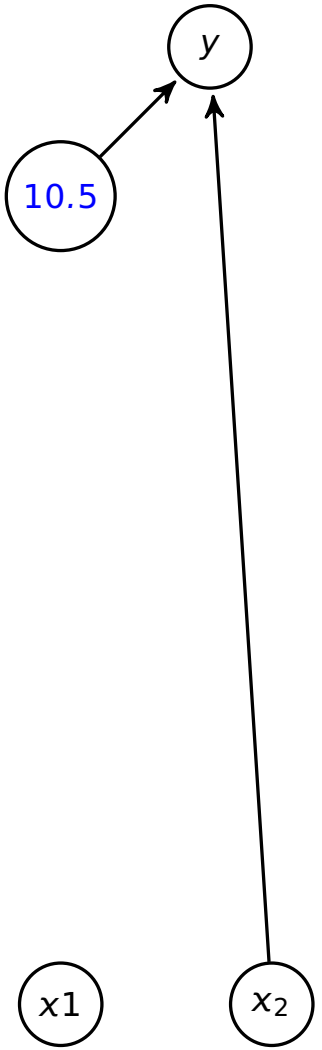
Insignificance test:

$$S_y(v_i) \leq \epsilon$$

Significance



Optimised DAG



Significance Analysis in a nutshell (IV)

Input data:

$x1, x2 \in [1, 36000]$
$\epsilon = 2.5$

Original code:

```
v1 = x1 * x2;
v2 = sin(x1);
v3 = exp(v2);
v4 = v3 * v1;
v5 = log( v4 );
y  = v2/10 + x2/100;
```

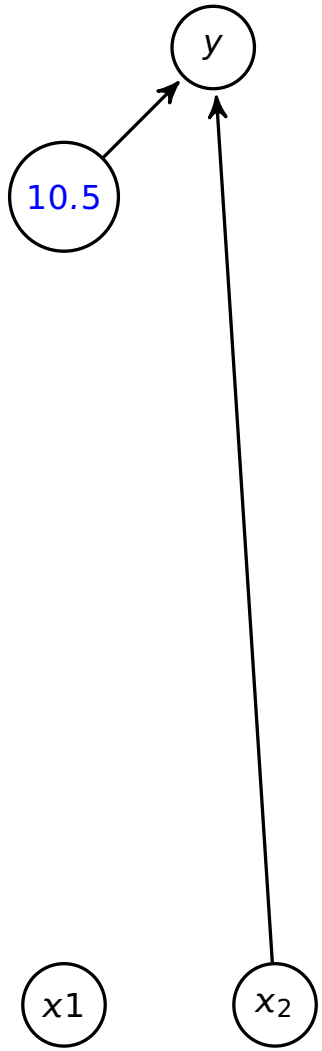
$\longrightarrow y \in [-0.09, 362]$

Approximating code:

```
v5 = 10.5;
y = v5/10 + x2/100;
```

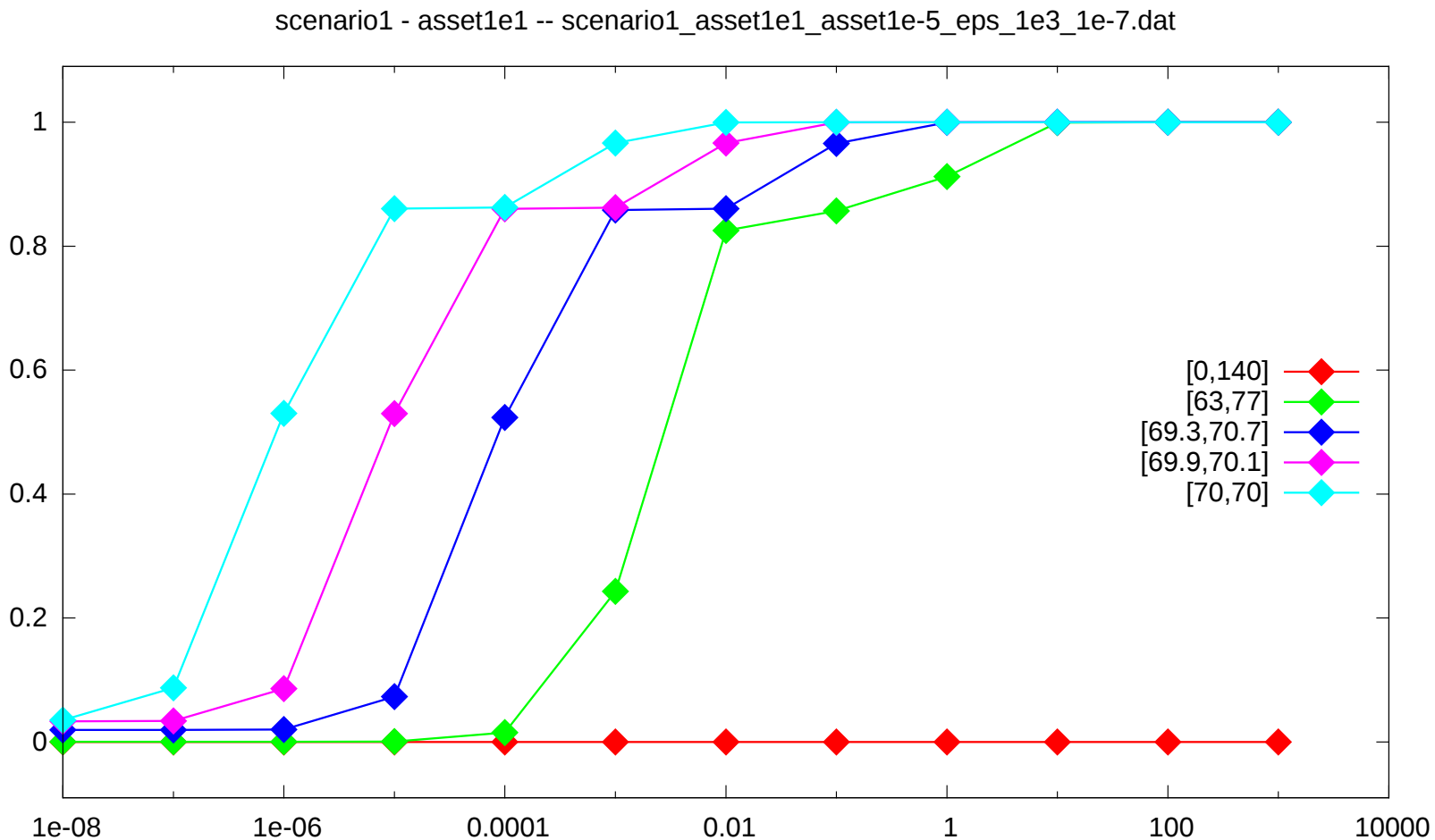
$\longrightarrow y \in [1.06, 361]$

Optimised DAG



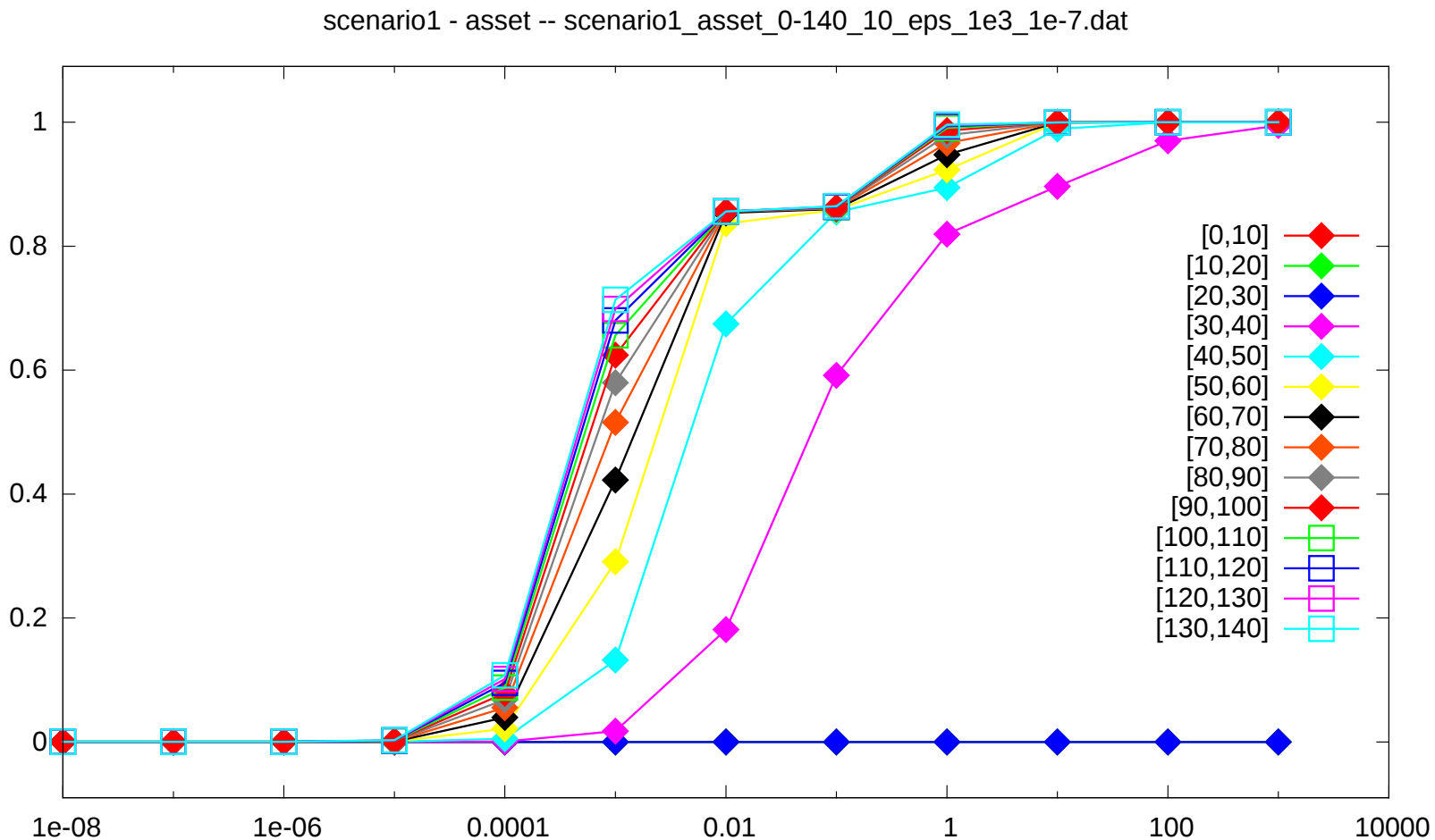
Hello Finance (Monte Carlo), Increasing input intervals

Percentage of insignificant program variables.



Hello Finance (Monte Carlo), Input interval cover

Percentage of insignificant program variables.



Hello Finance (PDE), in-house CFD flow solver

Interval - values and -adjoints even for point inputs : Complete $\boldsymbol{R}$

```
##################################################
        HelloFinance/PDE
##################################################

Price: [ ENTIRE ]
d_Price/d_K: [ ENTIRE ]
d_Price/d_S0: [ ENTIRE ]
d_Price/d_r: [ ENTIRE ]
```

We debugged that code to understand where the problem arises:

- No specific operation, constant growth
- At some point program variables became big and bigger . . .

  $\Longrightarrow$ wrapping effect (well known / feared)

Outlook

Outlook

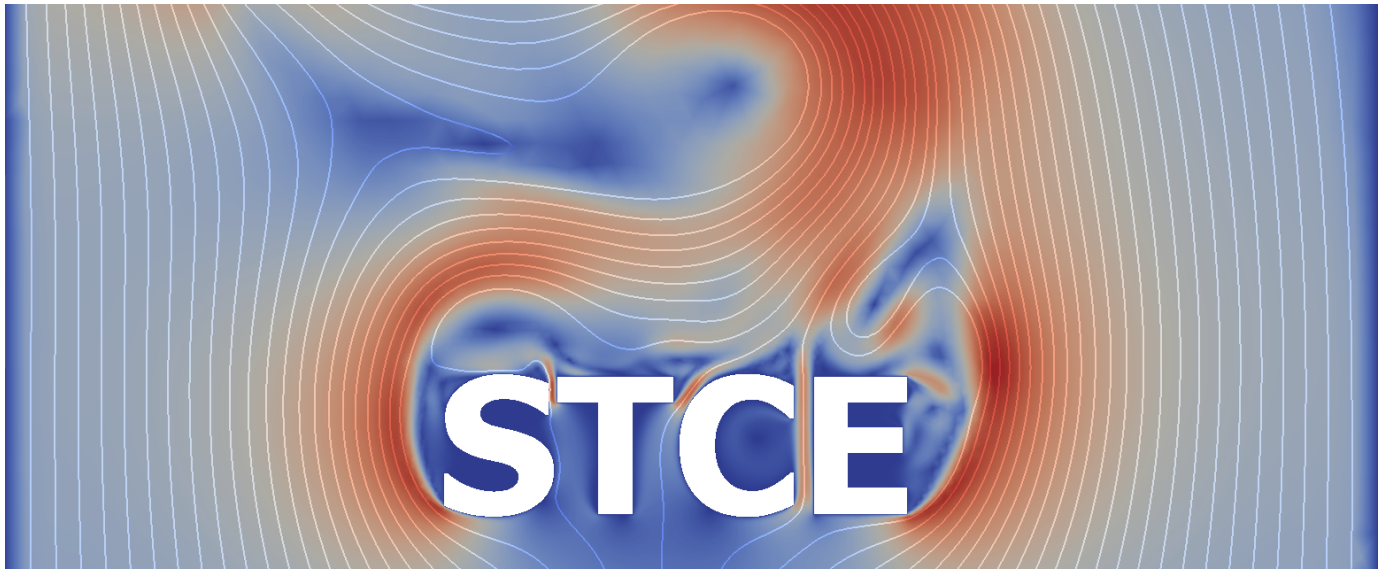
Goal: General purpose automatic significance analysis

We are working on an algorithm based on parallel recursive interval splitting of value intervals of program variables that will handle

- not feasible relational operators (ensure **control flow feasibility**),
- overestimation of value intervals (ensure **value feasibility**),
- overestimation of adjoint intervals (ensure **adjoint feasibility**).

Issues to address:

- More understanding of significance criteria is needed.
- New (scaled) significance criteria will be explored.
- How to rank feasible scenarios coming from the automatic significance analysis?
- What interactions exist between *control flow feasibility*, *value feasibility*, and *value feasibility*?
- How to split value intervals?
- How to schedule the parallel analysis processes efficiently?



Thank you very much!

www.stce.rwth-aachen.de
[naumann,riehme]@stce.rwth-aachen.de

Resources

- **SCoRPiO deliverable D1.1** : Jan Riehme and Uwe Naumann. SCoRPiO Deliverable D1.1: Significance Based Computing Modeling. `www.scorpio-project.eu/wp-content/uploads/2014/07/Scorpio_D1.1.pdf`, June 2014.
- **`dco/c++`** : Software and Tools for Scientific Engineering, RWTH Aachen University, Germany. Derivative Code by Overloading in C++ (dco/c++). `http://www.stce.rwth-aachen.de/software/dco_cpp.html`.
- **`filib++`** : Michael Lerch, German Tischler, J"urgen Wolff von Gudenberg, Werner Hofschuster, and Walter Kr"amer. FILIB++ interval library. `www2.math.uni-wuppertal.de/~xsc/software/filib.html`.
- **Algorithmic Differentiation** :
  - A. Griewank and Andrea Walther. Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation. SIAM, 2nd edition, 2008.
  - U. Naumann. The Art of Differentiating Computer Programs. An Introduction to Algorithmic Differentiation. Software, Environments, and Tools. SIAM, 2011.
  - Community portal: `www.autodiff.org`
- **Interval Arithmetic** :
  - Ramon E. Moore. Interval Analysis. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1966.
  - Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. Introduction to Interval Analysis. Society for Industrial and Applied Mathematics, 1 edition, 1 2009.