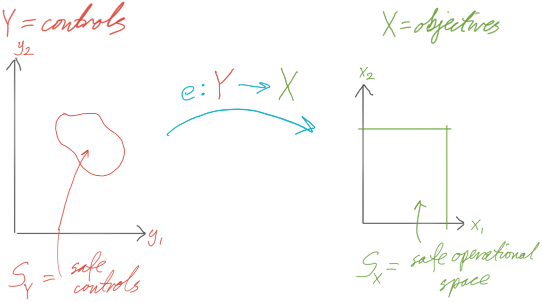
Domain-Specific Languages for Societal Challenges

Multi-objective optimisation and exploration of system simulations

Patrik Jansson

Functional Programming unit, Chalmers University of Technology



DSL → δσλ
DSLsofMath

Patrik Jansson, background

Physics
BSc

Mathematics
MSc

Comp Sci
PhD

Domain-
Specific
Languages

DSL → δσλ
DSLsofMath

Patrik Jansson: goals

- long term goal: create systems (theories, programming languages, libraries and tools) which make it easy to develop **reusable software components** together with proofs of their correctness.

Patrik Jansson: goals

- long term goal: create systems (theories, programming languages, libraries and tools) which make it easy to develop **reusable software components** together with proofs of their correctness.
- "software component" could be "agent-based model"

$DSL \to \delta\sigma\lambda$

DSLsofMath

Patrik Jansson: goals

- long term goal: create systems (theories, programming languages, libraries and tools) which make it easy to develop **reusable software components** together with proofs of their correctness.
- "software component" could be "agent-based model"
- ...expressed in a Domain-Specific Language (for ABMs)

What is a Domain Specific Language (DSL)?

- A DSL is an abstraction of a particular domain, supporting a domain specialist in building a model.
- An expression in a DSL can be seen as a formalised notion, a program, or a specification.
- Such expressions can often be executed, but also analysed as structured data.

What is a Domain Specific Language (DSL)?

- A DSL is an abstraction of a particular domain, supporting a domain specialist in building a model.
- An expression in a DSL can be seen as a formalised notion, a program, or a specification.
- Such expressions can often be executed, but also analysed as structured data.

DSL examples:

- Mathematics: Euclidean geometry — a DSL about points, lines, circles, etc.; or Arithmetic expressions (like $(x + y)/2$)
- Tools: Excel, General Algebraic Modeling System (GAMS), LexiFi (financial contracts), ...

$DSL \to \delta\sigma\lambda$

DSLsofMath

My work on Societal Challenges: Climate

2007–: work with Potsdam Inst. for Climate Impact Research (PIK)

- PIK wanted correct implementations of models for simulating global systems (both for economy and climate)
- I knew *Algebra of Programming*, and a long-term collaboration started
- with Cezar Ionescu, Nicola Botta, Carlo Jaeger, and Sarah Wolf
- Agent-based models early on, specified in TEX, Haskell, Agda

My work on Societal Challenges: Climate

2007–: work with Potsdam Inst. for Climate Impact Research (PIK)

- PIK wanted correct implementations of models for simulating global systems (both for economy and climate)
- I knew *Algebra of Programming*, and a long-term collaboration started
- with Cezar Ionescu, Nicola Botta, Carlo Jaeger, and Sarah Wolf
- Agent-based models early on, specified in TeX, Haskell, Agda

Agenti segreti 003

cj, May 17, 2009

The present document defines specifications for a program representing three agents. Each agent is a dynamical system with discrete time, as is the system as a whole.

Agents

The agents are:

- a firm (with state space F and observable set O_F),
- a working household (with state space H_w and observable set O_w),
- a wealthy household (with state space H_r and observable set O_r).

…

In the transition functions (and the functions used as components therein), the following parameters are used :

$$\begin{aligned} \alpha, \nu, \rho &\in (0,1) \qquad (7)\\ a, b, c &\in \mathbb{R}_{>1}\\ \delta &\in [0,1]\\ \hat{r}, \chi, \zeta, \hat{v} &\in \mathbb{R}_{>0} \end{aligned}$$

The following functions are used as components of transition functions:

$$\begin{aligned} f_k : K \times L \times Z \times W \to K \quad & f_k(k,l,z,w) = k \cdot (1-\delta) + f_y(k,l,z) - l \cdot w \qquad (8)\\ & \text{for } k \cdot (1-\delta) + f_y(k,l,z) > l \cdot w,\\ & \text{Nothing otherwise,}\\ f_y : K \times L \times Z \to J \quad & f_y(k,l,z) = k^{\alpha} \cdot (l \cdot z)^{1-\alpha} \text{ for } k > 0,\\ & \text{Nothing otherwise}\\ f_l : L \times L \to L \quad & f_l(l,l') = \min\,(l,\; l')\\ f_w : W \times \mathbb{R} \to W, \quad & f_w(w,r) = w \cdot \frac{a}{1 + \mathbf{e}^{b \cdot (\hat{r} - r)}} \end{aligned}$$

…

My work on Societal Challenges: GSDP

- Workshop on "Domain Specific Languages for Economical and Environmental Modelling (DSL4EE)" 2011, Marstrand, Sweden.
- developed domain-specific high-level concepts for **multi-agent modelling**, **sequential decision problems** and for computational **vulnerability** assessment
- successfully applied software specification to **economy theory** and to **climate impact research**
  - *Dependently-Typed Programming in Scientific Computing - Examples from **Economic Modelling***, Ionescu & Jansson, 2013
  - ***Testing versus proving** in climate impact research*, Ionescu & Jansson, 2013

Funded by Global Systems Dynamics and Policy (GSDP),
FP7 FET-Open project (1.3M EUR, 2010–13)

Global Systems Science

Selected publications (cont.):

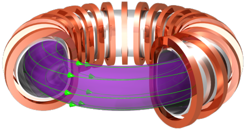
- ***Domain-Specific Languages of Mathematics****: Presenting Mathematical Analysis using Functional Programming*, 2016
- ***Sequential decision problems****, dependent types and generic solutions*, 2017
- *Contributions to a computational theory of* ***policy advice*** *and avoidability*, 2017
- *The impact of* ***uncertainty*** *on optimal emission policies*, 2018

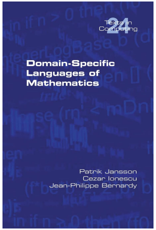
Funded by

- Global systems Rapid Assessment tools through Constraint FUnctional Languages (GRACeFUL, 2.4M EUR, 2015–2017)
- Centre of excellence for Global Systems Science (CoeGSS), 4.5M EUR, 2015 – 2018

- Project: OptiFun: Optimising Fusion with Generative Programming, 3M SEK, 2022–23
- *Domain-Specific Languages of Mathematics*, College Publications, 2022
- ***Responsibility Under Uncertainty**: Which Climate Decisions Matter Most?*, 2022 (under review)

Related project: Tipping Points in the Earth System (TiPES, 8.5M EUR, 2019–2024)





Optimisation example: Part 1

The object of study is an expensive, black-box function
$e : Y \to X$ where we have controls $Y = \mathbb{R}^n$ and observations $X = \mathbb{R}^m$

Optimisation example: Part 1

The object of study is an expensive, black-box function $e : Y \rightarrow X$ where we have controls $Y = \mathbb{R}^n$ and observations $X = \mathbb{R}^m$

In the simplest form (and with $m = 1$), the aim is to just find an input-output-pair (y^*, x^*) at a (global) minimum of e:

$x^* = e\,(y^*)$ and $\forall\ y : Y.\ \ x^* \leqslant e\,(y)$.

This search can be done using Bayesian optimisation or with other optimisation techniques.

Optimisation example: Part 1

The object of study is an expensive, black-box function $e : Y \to X$ where we have controls $Y = \mathbb{R}^n$ and observations $X = \mathbb{R}^m$

In the simplest form (and with $m = 1$), the aim is to just find an input-output-pair (y^*, x^*) at a (global) minimum of e:

$x^* = e\,(y^*)$ and $\forall\ y : Y.\ \ x^* \leqslant e\,(y)$.

This search can be done using Bayesian optimisation or with other optimisation techniques.

Unfortunately, it is in general impossible to implement this specification: as the set $Y = \mathbb{R}^n$ is infinite, for any finite number of points evaluated, there is always the risk that some other point is better.

Optimisation example: Part 1

The object of study is an expensive, black-box function $e : Y \to X$ where we have controls $Y = \mathbb{R}^n$ and observations $X = \mathbb{R}^m$

In the simplest form (and with $m = 1$), the aim is to just find an input-output-pair (y^*, x^*) at a (global) minimum of e:

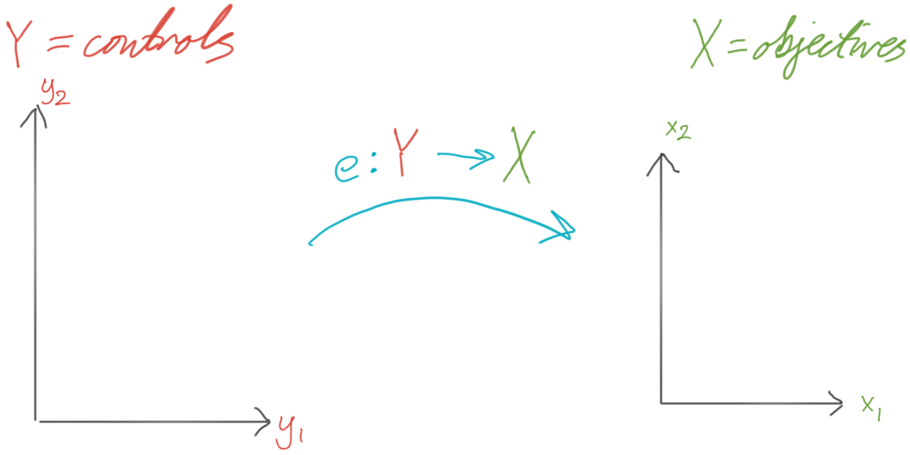
$x^* = e\,(y^*)$ and $\forall\; y : Y.\;\; x^* \leqslant e\,(y)$.

This search can be done using Bayesian optimisation or with other optimisation techniques.

Unfortunately, it is in general impossible to implement this specification: as the set $Y = \mathbb{R}^n$ is infinite, for any finite number of points evaluated, there is always the risk that some other point is better.

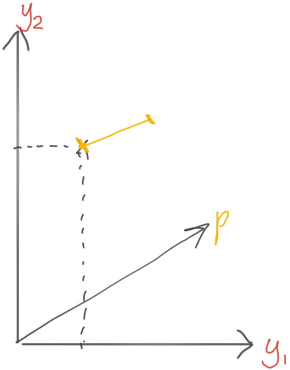
Fortunately, if we require a certain degree of smoothness of e (continuous, with concrete bounds on its derivatives), and if settle for a "good" point (within some ϵ of the true optimum), it becomes implementable.

Optimisation example: Part 1, figure



Optimisation example: Parameters & uncertainty

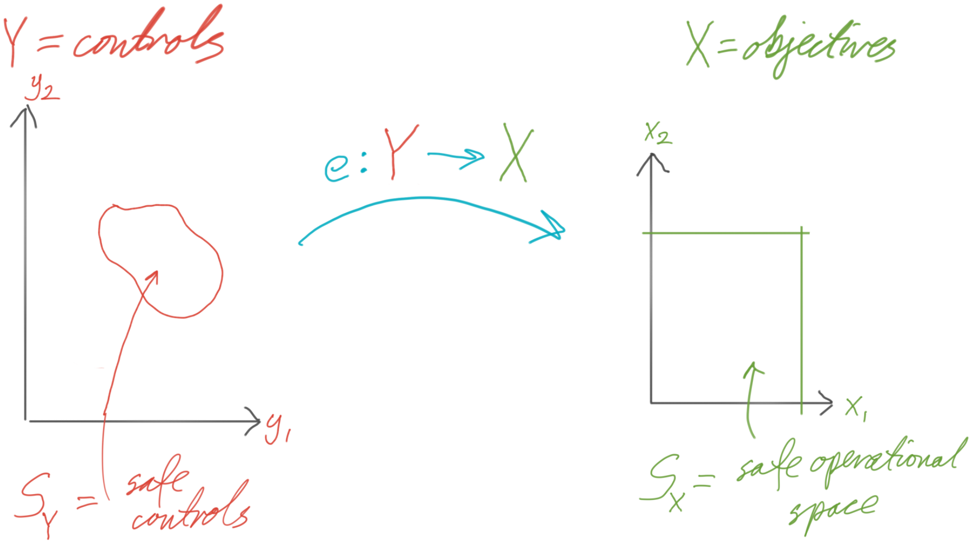
Y = controls P = parameter space X = objectives



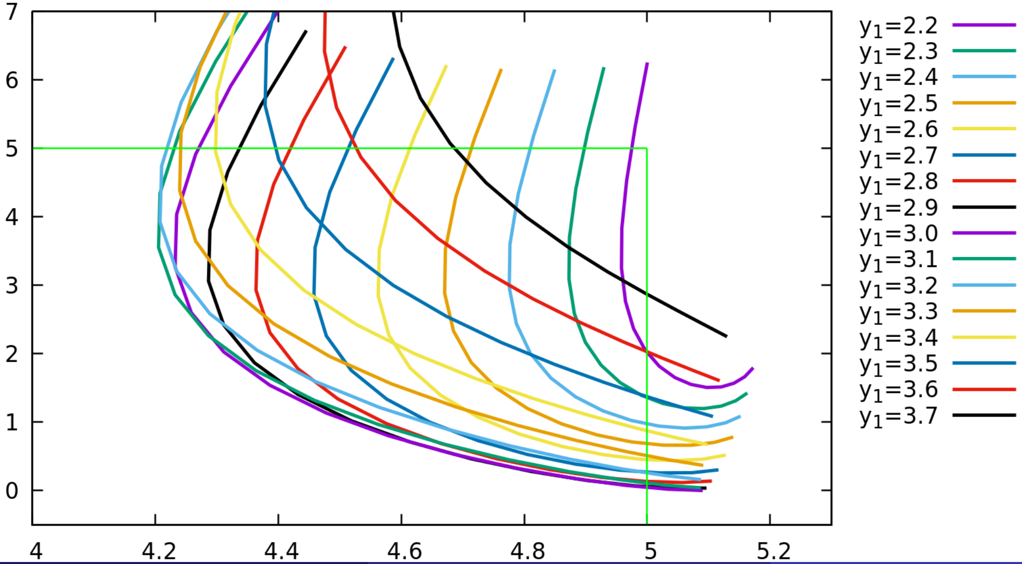
$e : P \times Y \to X$

sensitivity
stability
uncertainty

Safe controls / safe operational space



Numeric simulation example of safe operational space



Optimisation: Part 3: multi-objective exploration

In our intended application we have a *multi-objective* optimisation problem.

Optimisation: Part 3: multi-objective exploration

In our intended application we have a *multi-objective* optimisation problem.

The typical optimisation framework will insist on just one measurement which requires a way to "measure" (combine) the objectives. This is often ad-hoc and could be problematic (ethics, differing stakeholders, etc.)

Optimisation: Part 3: multi-objective exploration

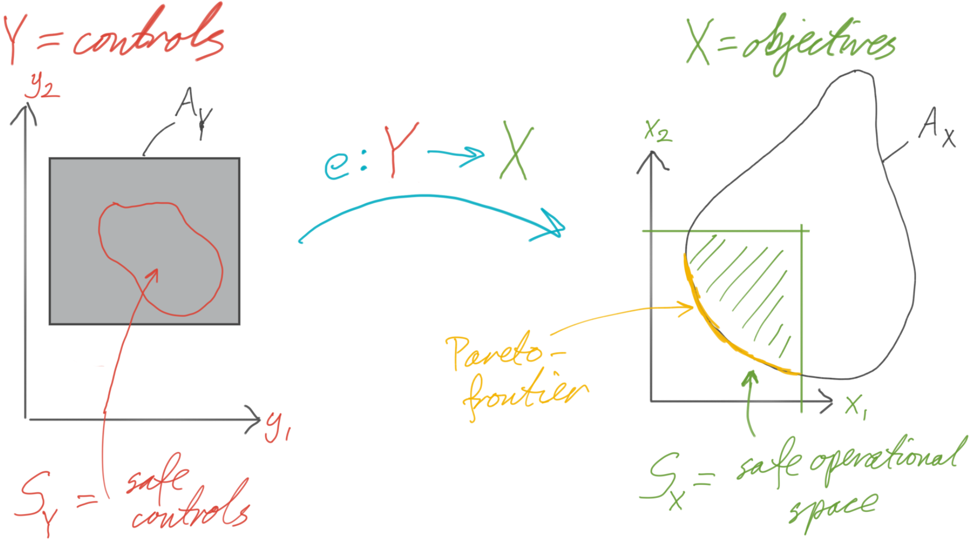
In our intended application we have a *multi-objective* optimisation problem.

The typical optimisation framework will insist on just one measurement which requires a way to "measure" (combine) the objectives. This is often ad-hoc and could be problematic (ethics, differing stakeholders, etc.)

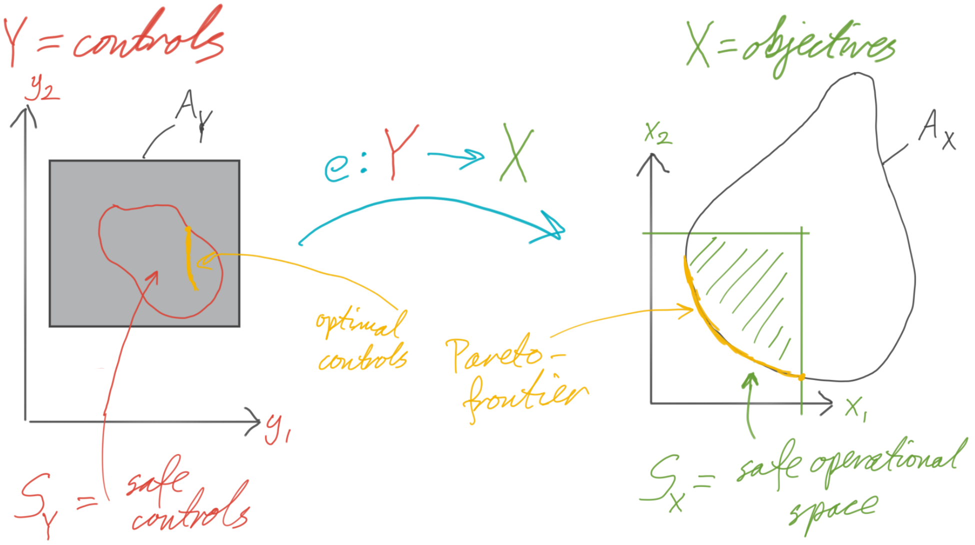
Our aim here is to visualise the objective space, illustrating the trade-off between the objectives (the Pareto frontier, etc.).

This allow users (stakeholders) to choose which is the best compromise.

Exploring controls / different kinds of optima

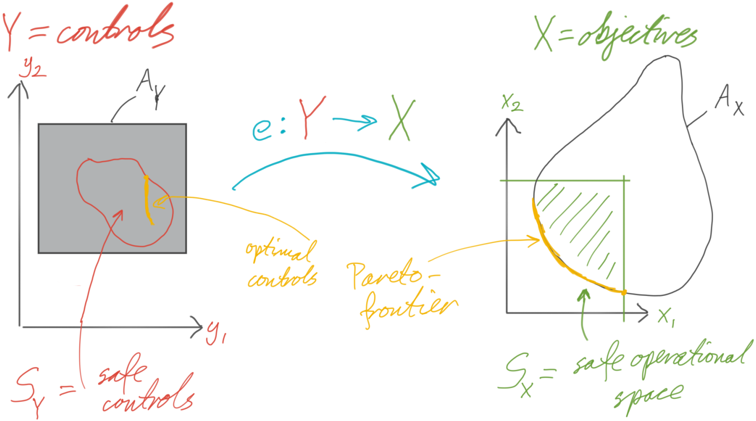


Pareto frontier + its inverse image



Domain-Specific Languages for Global Systems Science

- Multi-objective “optimisation” / exploration
- Challenges: uncertainty / curse of dimensionality
- Domain-Specific Languages for high-level modelling



CHALMERS
UNIVERSITY OF TECHNOLOGY

DSL → δσλ
DSLsofMath