PyData

Bratislava

GAP DATA INSTITUTE

TIM: Large-scale Forecasting for Energy in Julia

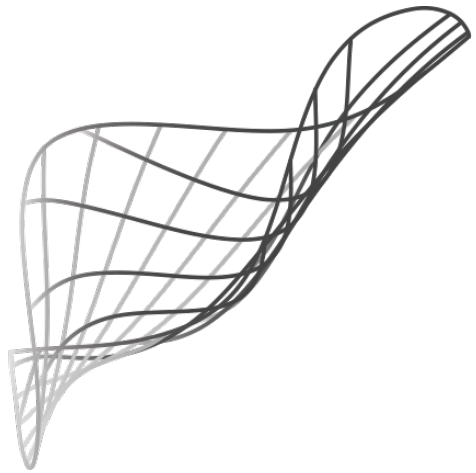
(PyData Bratislava Meetup #6, Nervosa)

19. 2. 2018

Ján Dolinský, Tangent Works #PyDataBA

TANGENT WORKS

ADVANCED FORECASTING

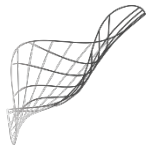


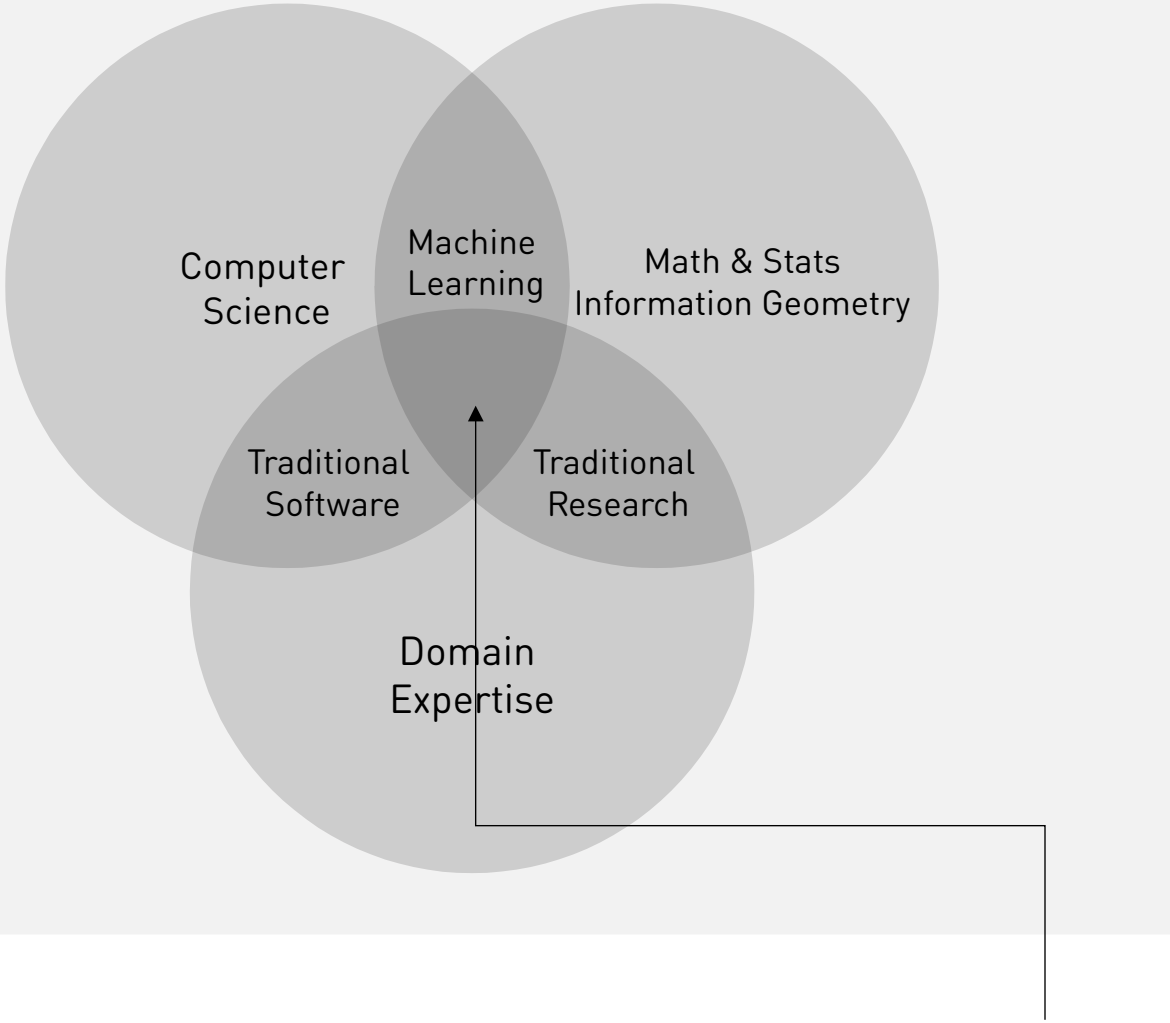
The Two Language Problem

The Philosofical Part

Ján Dolinský
jan.dolinsky@tangent.works

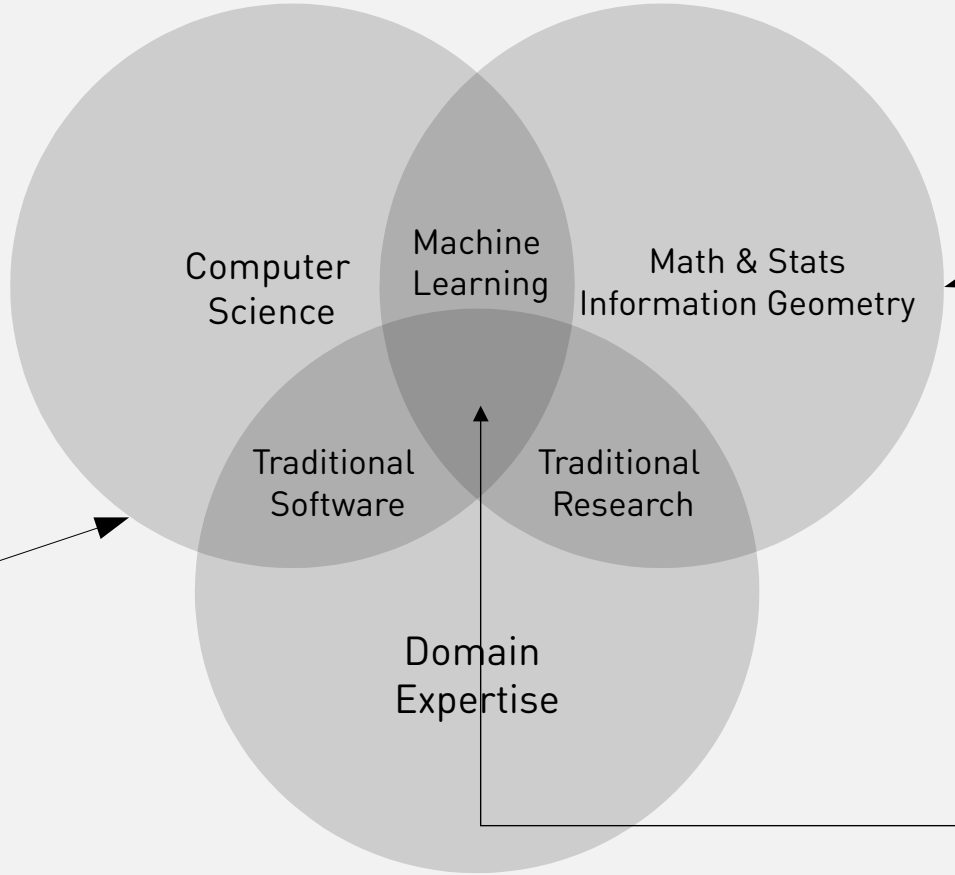
TANGENT WORKS
ADVANCED FORECASTING





TANGENT WORKS
ADVANCED FORECASTING

Data Science Stack:
R, Matlab, Python, Ruby, ...



Production Stack:
C++, Java, Fortran, ...

VALUE

TANGENT WORKS
ADVANCED FORECASTING

The Two Language Problem

Prototyping

Production

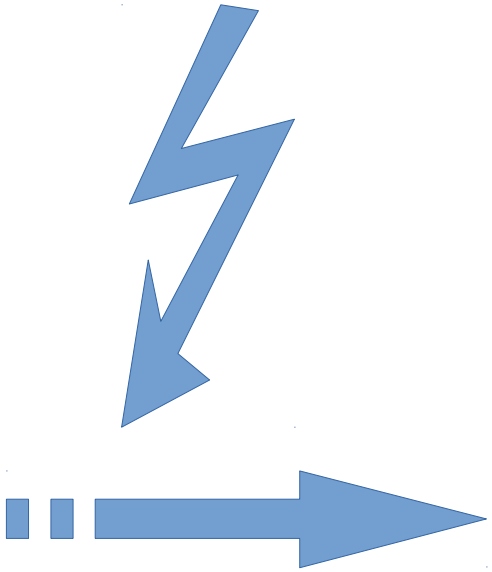
Mathematicians & Data Scientists
Getting the applied math right.

Performance engineers & Architects.

The Two Language Problem: Tangent Works in 2014

Prototyping

Production



The Data Science Stack:
R, Matlab, Octave, Python, Ruby, ...

Production stack:
C++, Fortran, Java, ...

A lot of effort ...

The Two Language Problem: Tangent Works in 2018

Prototyping

Production

Julia

Julia

Mathematicians
Getting the applied math right.

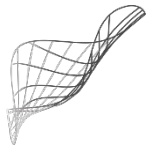
Performance engineers & Architects.

TIM: Automatic Model Building for Time-series in Energy

The Product Part

Ján Dolinský
jan.dolinsky@tangent.works

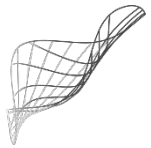
TANGENT WORKS
ADVANCED FORECASTING



TIM
Tangent Information Modeller

Ján Dolinský
jan.dolinsky@tangent.works

TANGENT WORKS
ADVANCED FORECASTING



Content

- Time-series Problems in Energy Industry
- Data Science Process and Automatic Model Building
- Live Modeling Demonstration
- Large-scale Forecasting Systems & Why Automation Matters
- Julia Language
- GefCom 2014, 2017 results & Summary

Time-series Problems in Energy Industry

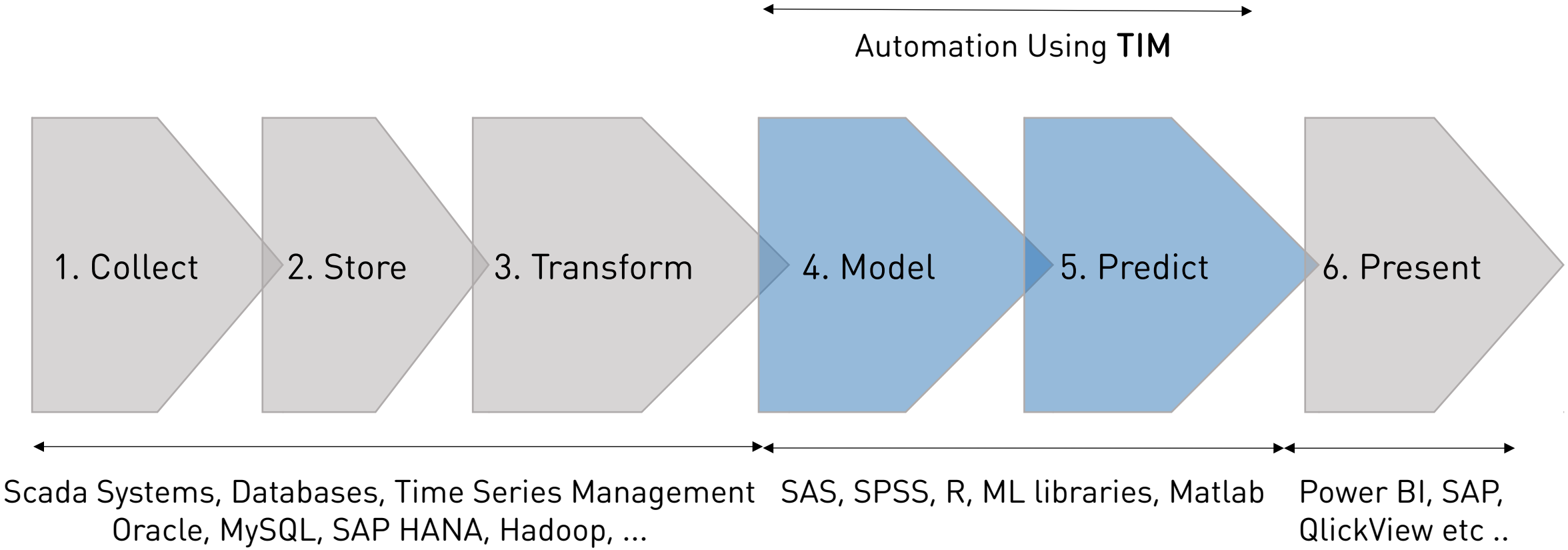
- Electricity Load (aggregated and individual)
- Technical Losses
- System Imbalance
- Gas Consumption (District Heating)
- District Cooling

Consumption side

- Wind Production
- Solar Production
- Electricity Price

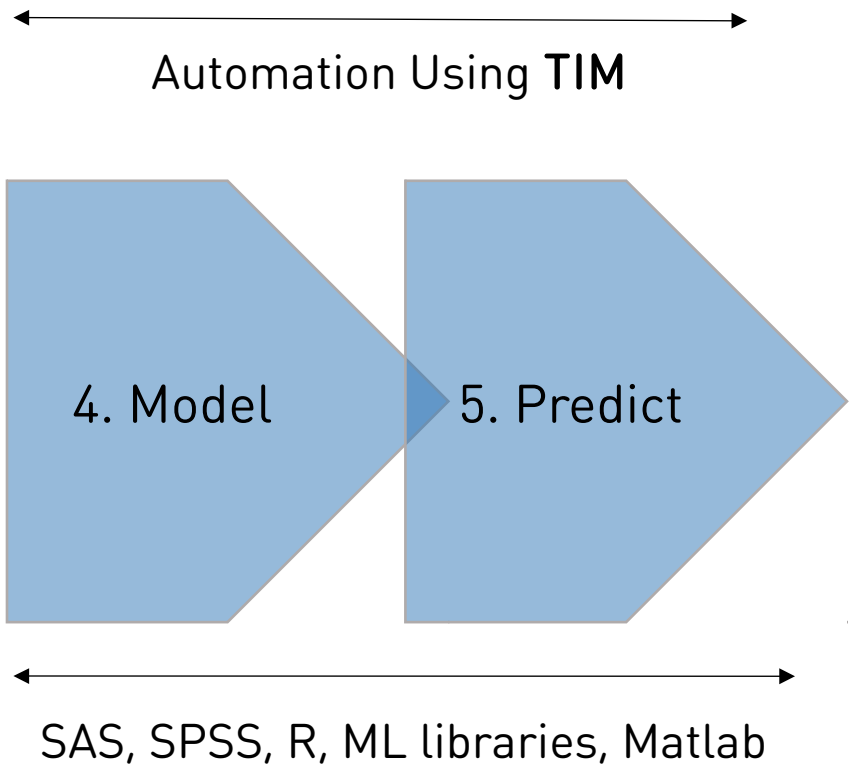
Production side

Data Science Process



Data Science Process

- feature engineering
- model selection
- tedious
- costs money



Data Science Process

Which features and how many ?

$$y(k) = f\left(Temp(k-3), Temp(k-22) * Wind(k-1), y(k-24)\right)$$

NN, SVM, MARS, LASSO, RF, ... ?

// optional

Data Science Process

$$y(k) = f\left(Temp(k-3), Temp(k-22) * Wind(k-1), y(k-24)\right)$$

Temp(k-1), Temp(k-2), ..., Temp(k-24)
Wind(k-1), Wind(k-2), ..., Wind(k-24)
y(k-1), y(k-2), ..., y(k-24)

24 + 24 + 24 + 24*24 = 24*27 = 648

Temp(k-1), Temp(k-2), ..., Temp(k-96)
Wind(k-1), Wind(k-2), ..., Wind(k-96)
y(k-1), y(k-2), ..., y(k-96)

96 + 96 + 96 + 96*96 = 96*99 = 9504

TIM: As a Large-Scale Model Building Engine

Business User Mode produces high-quality models.

TIM: As a Data Science Tool

Advanced User Mode allows to explore new scenarios quickly.
Fine-tuning possible for critical assets.

Live Demonstration of TIM

GefCom 2017 ex-ante results

- Qualifying Match: 1st place out of 177 teams
- Final Match: 2nd place out of 12 pre-selected teams

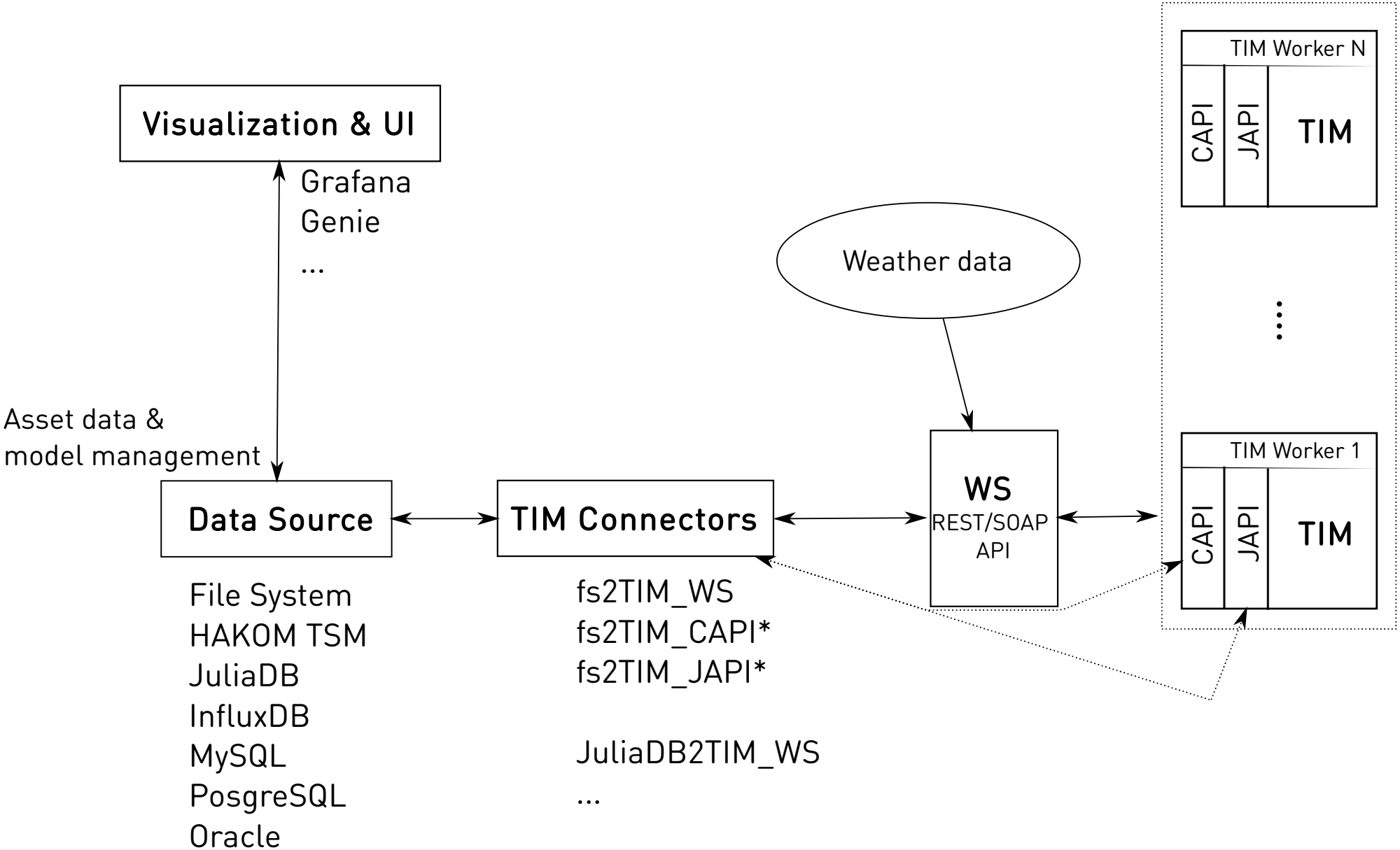
Andritz Hackathon 2017

1st place out of 7 ML companies

Large-scale Forecasting Systems

Platforms where thousands of different consumption and production models are produced.

This is not possible without full automation.


Visualization & UI
Grafana
Genie
...
Asset data &
model management
Data Source
File System
HAKOM TSM
JuliaDB
InfluxDB
MySQL
PosgreSQL
Oracle
TIM Connectors
fs2TIM_WS
fs2TIM_CAPI*
fs2TIM_JAPI*
JuliaDB2TIM_WS
...
Weather data
WS
REST/SOAP
API
TIM Worker N
CAPI
JAPI
TIM
TIM Worker 1
CAPI
JAPI
TIM

Julia Language

Walks like Python runs like C

- A single platform for prototyping and production → significant gains in efficiency of product development
- Vectorized code runs equally fast as de-vectorized
- Vectorization on a single thread level
  - SIMD out of the box
  - Direct calls to BLAS
- Distributed parallel computation (multiple threads)

Summary

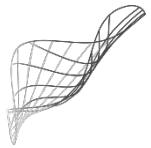
- Fully automated model building
- Accuracy often outperforms manually build models
- Quick insights into a time-series of interest
- Tedious model building is an option not necessity

Thank you.

jan.dolinsky@tangent.works

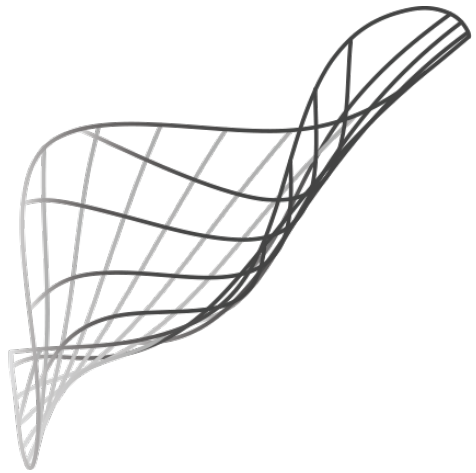
Tangent Works
Na Slavíne 1
81106 Bratislava
www.tangent.works

TANGENT WORKS
ADVANCED FORECASTING



TANGENT WORKS

ADVANCED FORECASTING

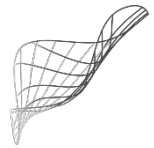


The Key Concepts in Julia

Julia Lectures for Tangent Works Employees

Ján Dolinský
jan.dolinsky@tangent.works

TANGENT WORKS
ADVANCED FORECASTING



Lectures Block 1: Vectorization vs. De-vectorization

- Iterators: The Storage Concept
- On Demand Iteration: Generators
- MapReduce principle
- Broadcasting
- Syntactic Loop Fusion
  - ✓ Why is vectorized code fast ?
  - ✓ Why it is not as fast as it could be ?

Lectures Block 2: Linear Algebra

- Matrices storage and its consequences
- Column-major & SIMD
- BLAS calls

Lectures Block 3: Language Design Elements

- Multiple-dispatch
- Struct
- Mutable & Immutable concept
- Modules & Packages generation
- JIT and AOT compilation

Julia Language

Walks like Python runs like C

- A single platform for prototyping and production → significant gains in efficiency of product development
- Vectorized code runs equally fast as de-vectorized
- Vectorization on a single thread level
  - SIMD out of the box
  - Direct calls to BLAS
- Distributed parallel computation (multiple threads)

2 Years of Using Julia

- It is a real programming language
- In v0.2 to v0.4 still in heavy development
- v0.5 and v0.6 are very useful; static compilation possible
- Prototyping and production development is done using the same platform
- IDE not fully matured but already quite useful
- Debugger