Realtime capable first principle transport modelling for tokamaks

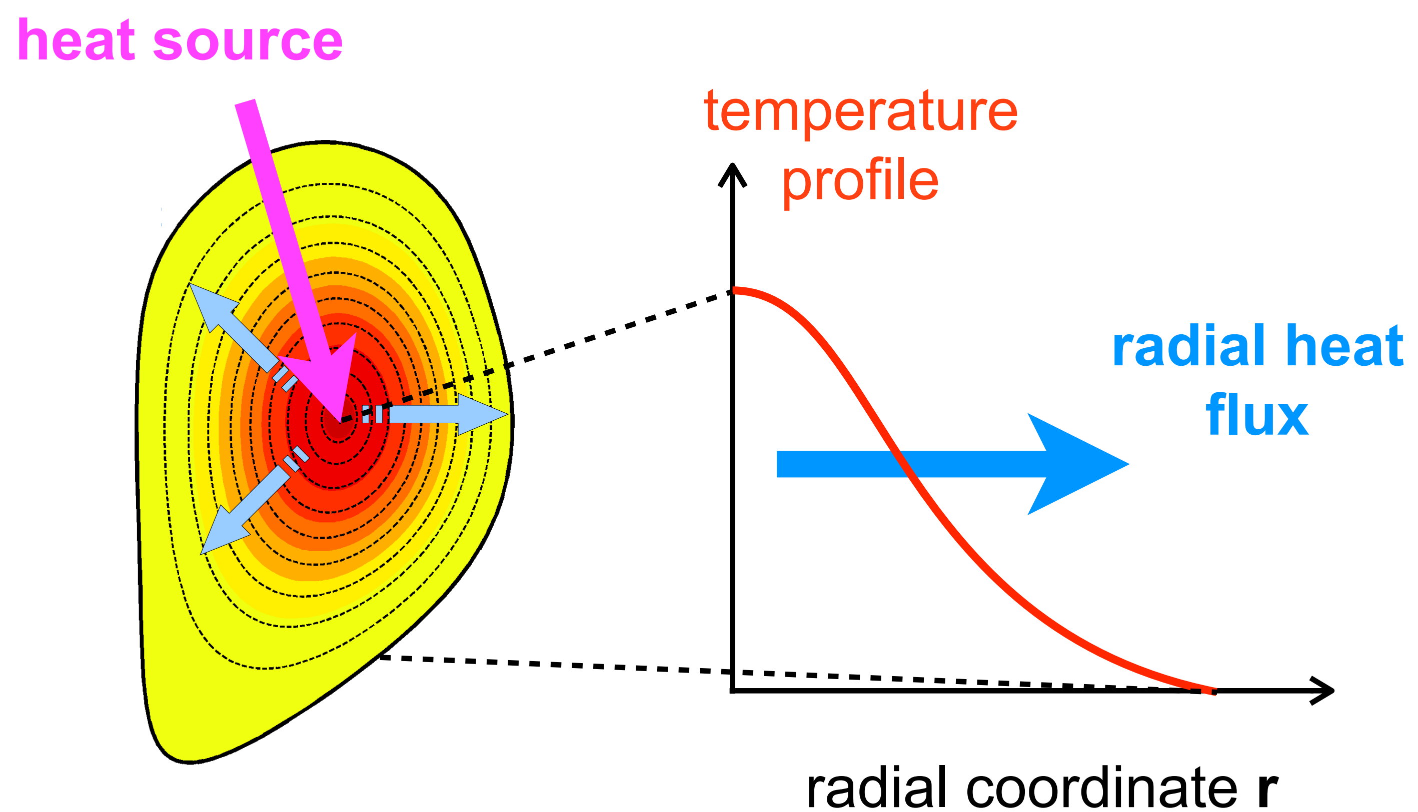
Y. Camenen[1]

T. Aniel[2], J.F. Artaud[2], C. Bourdelle[2], S. Breton[2], J. Citrin[3], F. Felici[4], J. Garcia[2], A. Ho[3], F. Imbeaux[2], R. Nouilletas[2]

[1] PIIM, CNRS/AMU, Marseille, France
[2] IRFM, CEA Cadarache, France
[3] FOM Institute DIFFER, Eindhoven, The Netherlands
[4] Eindhoven Univ. of Technology, Eindhoven, The Netherlands

Project aim

- Context: prediction of the temperature, density, and rotation profiles in tokamak plasmas
- Focus on **turbulent transport** → link between sources and profiles
- Example:



- Aim: **accurate** and **realtime** prediction of turbulent transport based on first principle models

Why fast(er) and first principle based?

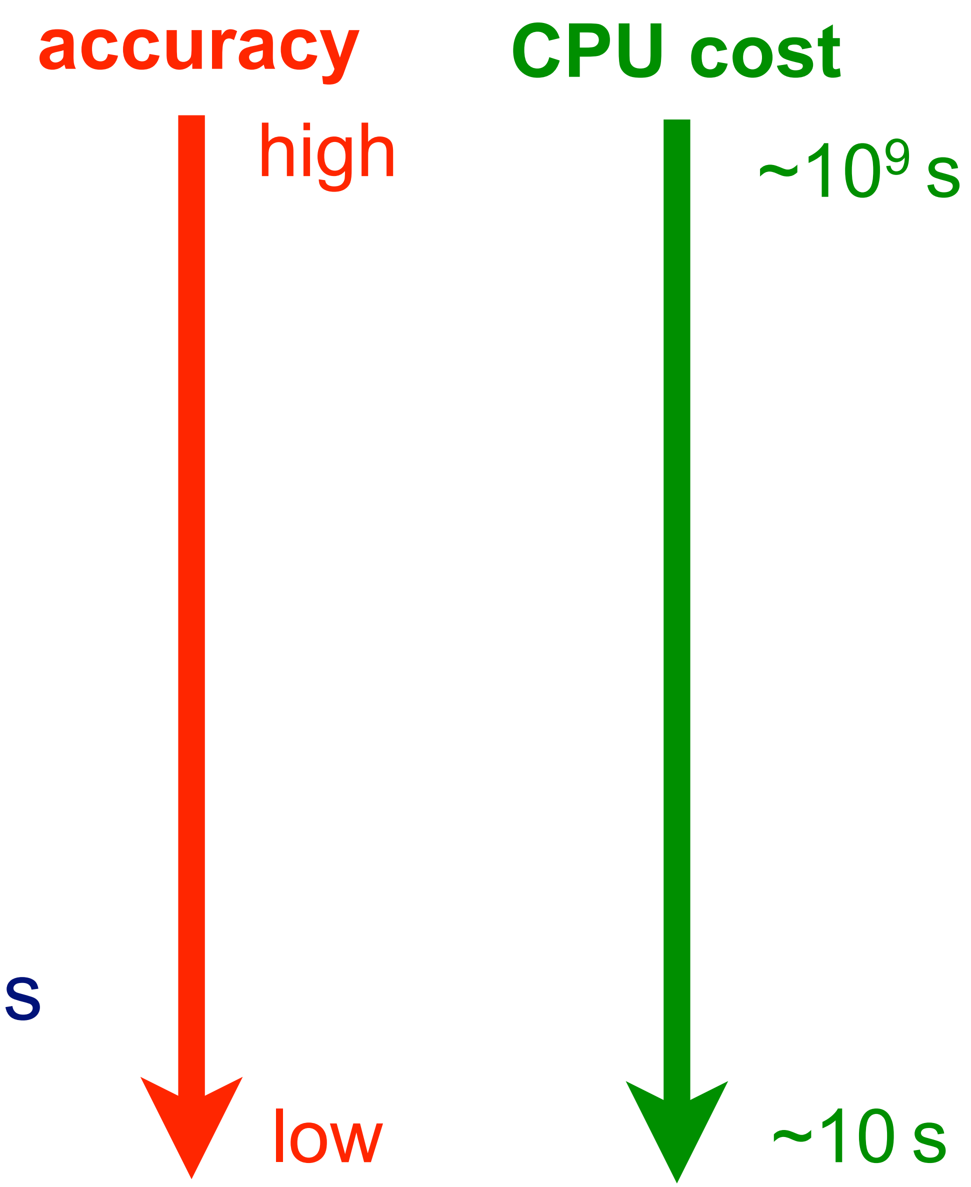
First principle based (and validated)

- Extrapolations to next step devices more reliable

Faster (<1ms on 1 CPU per flux computation)

- Real-time control applications
  - Model based controller design for profile control (e.g. RAPTOR)
  - Discharge supervision (disruption avoidance)
  - Discharge optimization on-the-fly
- Offline scenario development (trajectory optimisation)
- High throughput transport physics analysis
- Impact of measurements uncertainties
- Transport analyses between shots in the control room

Standard approach: model reduction



Non-linear gyrokinetic simulations
(local and gradient-driven)

Linear gyrokinetic simulations + quasi-linear saturation rule

Additional simplifications
QLK: gyro-kinetic with fluid eigenfunctions
TGLF: gyro-fluid model

- Simulation of 1s of JET plasma: about 1000 fluxes computation
- Still marginal for many applications → further speed-up required
- How to keep high physics fidelity??? A dead-end?

Use database regressions?

accuracy high → low

CPU cost ~10^9 s → ~10 s → < 1ms

Non-linear gyrokinetic simulations
(local and gradient-driven)

Linear gyrokinetic simulations + quasi-linear saturation rule

Additional simplifications
QLK: gyro-kinetic with fluid eigenfunctions
TGLF: gyro-fluid model

- Emulation from regressions of compiled databases
- Calculations relegated to pre-processing stage

A basic idea, but is it realistic?

Main issue: high dimensionality of the problem

- More than 20 input parameters per simulation
- With 5 values per dimension → 10^{14} points!!

Enough CPU time available? ✔

- 1 linear gyrokinetic simulation: 1 to 100 CPUh
- A 3 million entries database can be assembled in a few years
- Sufficient to cover the operational space of present devices exploiting the strong correlation between the inputs

Regression tools? ✔

- Multilayer neural network techniques are well suited
  - Used in climate modelling, same problem dimensionality
  - Mature technique, many existing libraries (Matlab, Python,...)

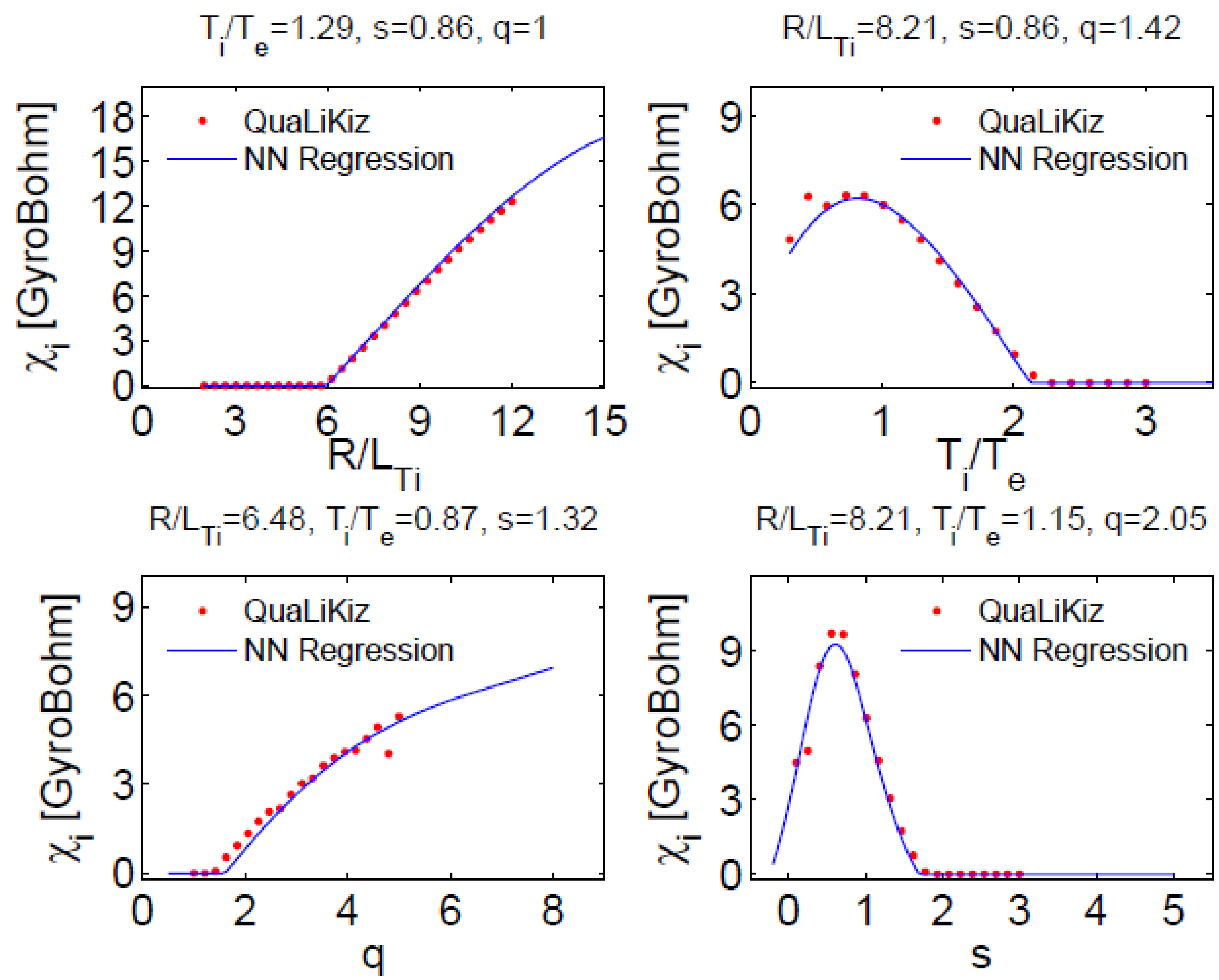
Proof of principle

Neural network fit for QuaLiKiz output. ITG regime

(S.Breton MSc, J. Redondo MSc ; Citrin, Breton *et al.,* Nucl. Fusion Lett. 2015)

5D input training set for ~50,000 fluxes

$q = 1-5\,;\ \hat{s} = 0.1-3\,;\ \frac{T_i}{T_e} = 0.3-3\,;\ \frac{R}{L_{Ti}} = 2-12\,;\ k_\theta\rho_s = 0.05-0.8$ (ion scales)



Parameter scans of NN ion heat conductivity vs original QuaLiKiz results

Note that regularization allows reasonable extrapolation.

Extrapolation not recommended, but encouraging for robustness in sparse datasets

[J. Citrin TTF Leysin 2016]

Roadmap

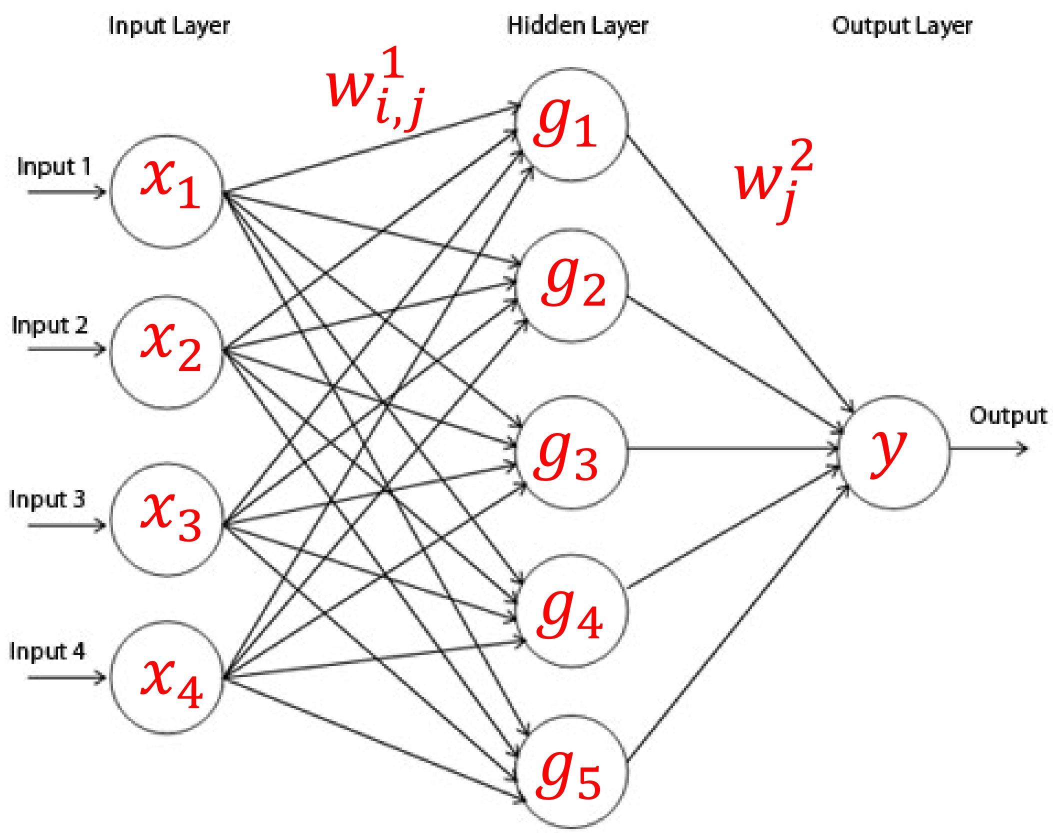
- **Proof of principle** - done
  - Neural network fit for Qualikiz output, 8D database, ITG regime
  - NN fit included in a transport code (CRONOS) and real-time control code (RAPTOR)
- **Set-up the infrastructure for the GK database** - done
  - Database content defined and documented
  - SQL database “online” with wrappers for inputs/outputs
- **Populate the database** - in progress
  - From 2015, computing time on Turing (IDRIS)
- **Train and refine neural network regressions**
- **Test control-oriented applications with RAPTOR**
- **Improve quasi-linear saturation rules**

Roadmap

- **Proof of principle** - done
  - Neural network fit for Qualikiz output, 8D database, ITG regime
  - NN fit included in a transport code (CRONOS) and real-time control code (RAPTOR)
- **Set-up the infrastructure for the GK database** - done
  - Database content defined and documented
  - SQL database “online” with wrappers for inputs/outputs
- **Populate the database** - in progress
  - From 2015, computing time on Turing (IDRIS)
- **Train and refine neural network regressions**
- **Test control-oriented applications with RAPTOR**
- **Improve quasi-linear saturation rules**

ER project
2017/2018
J. Citrin

Multilayer perceptron network (simple topology)



x: Inputs: e.g. T_i/T_e, q, $\hat{s}$, R/L_{ti}
y: Output: e.g. ion heat flux
$w^{1,2}$: free weights for optimization

$$y = \sum w_j^2 g_j\left(\sum w_{i,j}^1 x_i\right)$$

With, e.g. $g(x) = \dfrac{2}{1+e^{-x}} - 1$

Optimize weights by minimizing: $\sum_N (t_N - y_N)^2 + \lambda \sum \left(w_{ij}\right)^2$

t_N are target values, known from, e.g. QuaLiKiz runs
λ is the regularization factor. Critical for avoiding overfitting