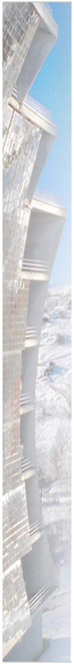


CANUM2026

Reconstruction of temperature and heat flux from Thermal Large Eddy Simulations

Yanis ZATOUT[1,2]

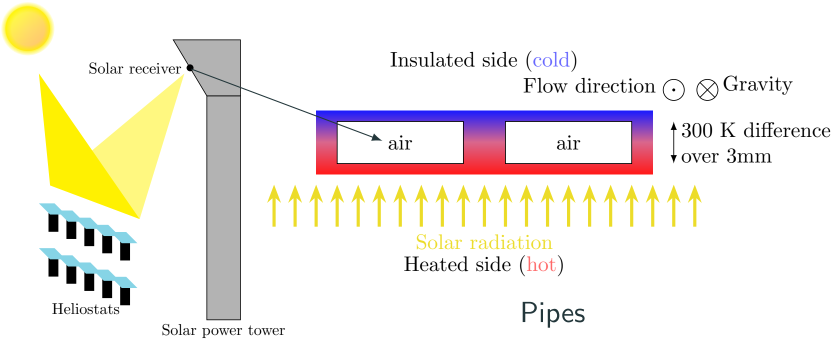
in colaboration with Onofrio SEMERARO[2], Lionel MATHELIN[2], Françoise BATAILLE[1], Adrien TOUTANT[1]

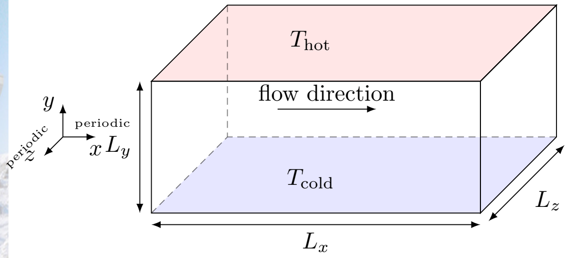
[1]PROMES-CNRS (UPR 8521), Université de Perpignan Via Domitia
[2]LISN-CNRS (UMR 9015), Université Paris-Saclay

4th of June 2026

Context

- Development of next generation concentrated **solar towers**
- **Solar receiver**: key component of solar towers
- Heats a heat-carrying fluid (**air**) using solar radiation





Geometry

- Simplest geometry representative solar receiver flows
- The flow is heated asymmetrically with fixed wall temperatures:

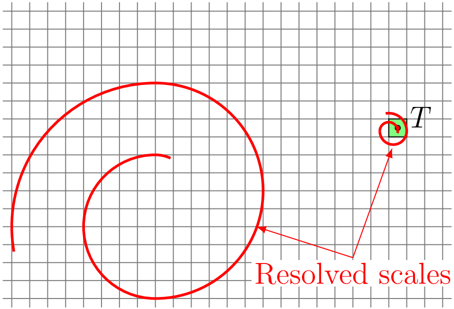
$$T_{\text{cold}} = 293\ \text{K},\ T_{\text{hot}} = 586\ \text{K}.$$

Discretization

- **Uniform mesh** in the periodic directions x and z.
- **Hyperbolic tangent** in the wall-normal direction y.

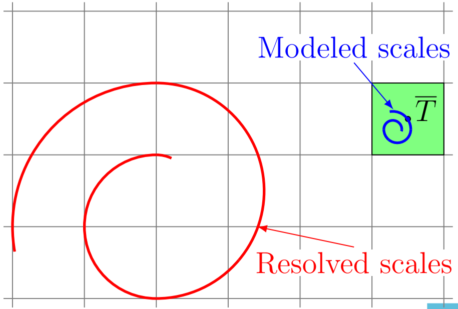
Direct Numerical Simulation

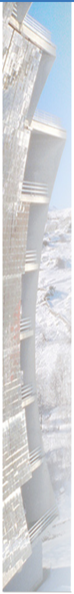
- All scales are resolved
- Very fine mesh
- Expensive to run



Thermal-Large Eddy Simulation

- Simulates large scales, models smallest
- Mesh is coarse → cheaper to simulate
- Simulation quantities are filtered ($\overline{T}$, $\overline{U}$,...)
- Requires closure modeling





Thermal-Large Eddy Simulations | Low-Mach number equations

- Mass conservation equation

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho U_j}{\partial x_j} = 0$$

- Momentum conservation equation

$$\frac{\partial \rho U_i}{\partial t} = -\frac{\partial \rho U_j U_i}{\partial x_j} - \frac{\partial P}{\partial x_i} + \frac{\partial \Sigma_{ij}(\boldsymbol{U}, T)}{\partial x_j}$$

- Energy conservation equation

$$\frac{\partial U_j}{\partial x_j} = -\frac{1}{\gamma P_0}\left((\gamma - 1)\frac{\partial Q_j(T)}{\partial x_j} + \frac{\mathrm{d}P_0}{\mathrm{d}t}\right)$$

- Ideal gas law

$$T = \frac{P_0}{\rho r}$$

- Mass conservation equation

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial \bar{\rho} \widetilde{U_j}}{\partial x_j} = 0,$$

- Momentum conservation equation

velocity-velocity closure

$$\frac{\partial \bar{\rho} \widetilde{U_i}}{\partial t} = -\frac{\partial \left(\bar{\rho} \widetilde{U_j} \widetilde{U_i} + \bar{\rho} \boxed{G_{U_j U_i}} \right)}{\partial x_j} - \frac{\partial \bar{P}}{\partial x_i} + \frac{\partial \Sigma_{ij}(\widetilde{\boldsymbol{U}}, \widetilde{T})}{\partial x_j},$$

- Energy conservation equation

$$\frac{\partial}{\partial x_j} \left(\widetilde{U_j} + \bar{\rho} \boxed{G_{U_j/\rho}} \right) = -\frac{1}{\gamma P_0} \left((\gamma - 1) \frac{\partial Q_j(\widetilde{T})}{\partial x_j} + \frac{\partial P_0}{\partial t} \right),$$

velocity-density closure

- Ideal gas law

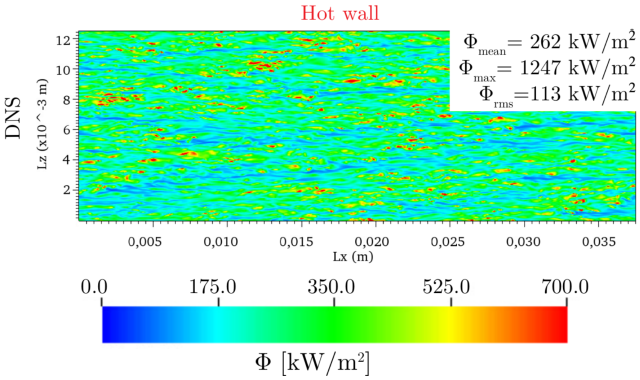
$$\widetilde{T} = \frac{P_0}{\bar{\rho} r},$$

Remark: $\bar{\rho} G_{U_j/\rho} = G_{U_j T} / \widetilde{T}$

Reconstruction

- Local heat flux imprecise
- Needs reconstruction techniques to recover temperature statistics
- Development of deconvolution techniques to accurately reconstruct fields
- The LES filter is **unknown**

Aim: Reconstruct **a posteriori**



Reconstruction

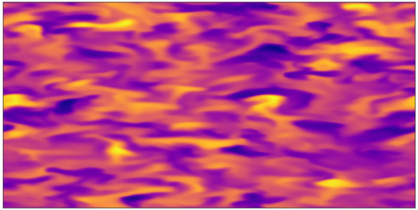
- Learning of an inversion operator for filtering on the temperature field:

$$T_{\mathrm{DNS}} \approx G^{-1} * T_{\mathrm{LES}}$$

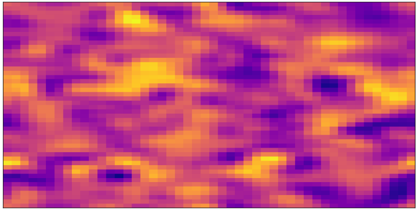
- The neural network learns the correction on the LES field

$$T_{\mathrm{pred}} = T_{\mathrm{LES}} + f_{\mathrm{CNN}}(T_{\mathrm{LES}}) \approx T_{\mathrm{DNS}}$$

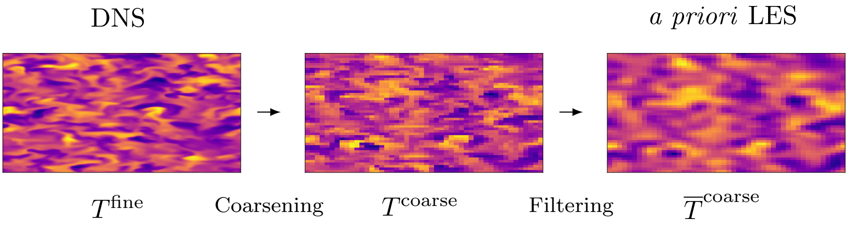
DNS



T-LES



Filtering process



- Anisothermal **DNS**, mean friction Reynolds number $Re_\tau = 180$, and Prandtl $Pr = 0.76$ after statistical convergence
- Timesteps spaced by $\Delta_t^+ = 7.76 \times 10^{-3}$
- Interpolate from a fine mesh of (384, 384, 266) points to (48, 48, 52) points
- Filter using a weighted top-hat filter

Reconstruction via Machine Learning | Wall-Normal Padding

The Problem Wall-normal direction (y) **not** periodic $\rightarrow$ requires boundary padding

Methodology

- Replicate boundary temperature outside domain
- Assume constant cell size at the boundary

Impact on Flow Data

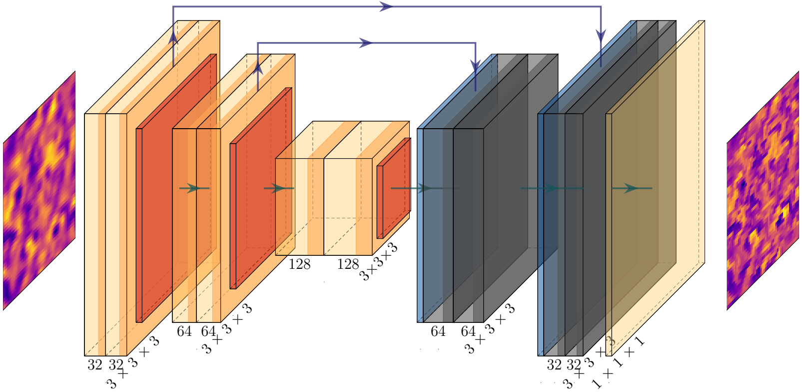
- **Preserved:** Second-order temperature statistics $\sqrt{\langle T'^2 \rangle}$.
- **Altered:** Boundary temperature gradient $Q_y = \lambda \frac{\partial T}{\partial y}$
- **Unaffected:** Sufficiently far from boundary

Benchmark

- We compare the performance of our model to an already existing method developed by Van Cittert. The inverse of a convolution filter G assumed invertible, writes

$$G^{-1} = (\mathcal{I} - (\mathcal{I} - G))^{-1}, \\ = \lim_{p \to \infty} \sum_{i=0}^{p} (\mathcal{I} - G)^i.$$

$$G_2^{-1} = \mathcal{I} - 2G + G^2, \\ G_3^{-1} = \mathcal{I} - 3G + 3G^2 - G^3,$$


32
32
3 + 3
64
64
3 + 3
128
128
3×3×3
64
64
3 + 3
32
32
3 + 3
1 + 1 + 1

Constrained learning task

- Method uses **a single** convolution filter **recursively**

$$T_r = \left(\sum_{n=0}^{5} (\mathcal{I} - G(\theta))^n \right) * \overline{T},$$

- The filter is constrained to be **conservative** during the inference step

$$\sum_{i,j,k} G_{i,j,k} = 1,$$

- The filter is constrained to be **isotropic** in the periodic directions

$$\forall (i,j), \qquad G_{i,j,k} = G_{j,i,k},$$

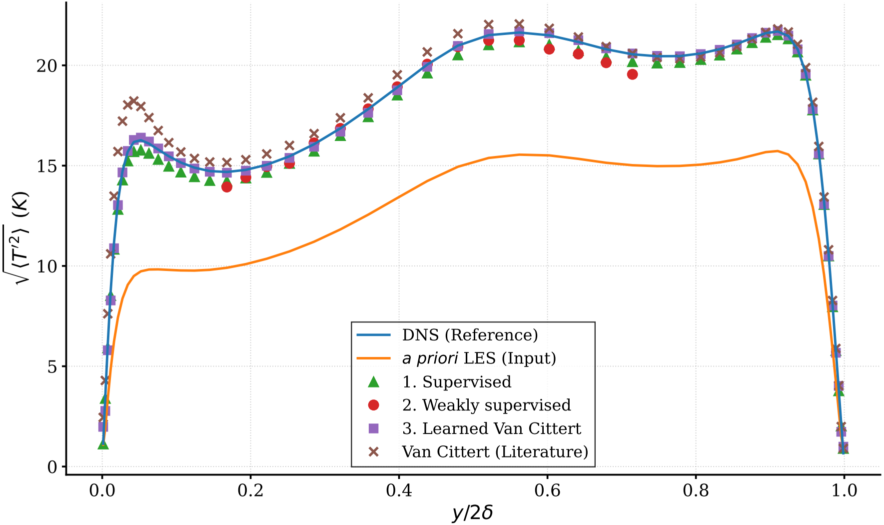
- Loss function is based on the second order statistic

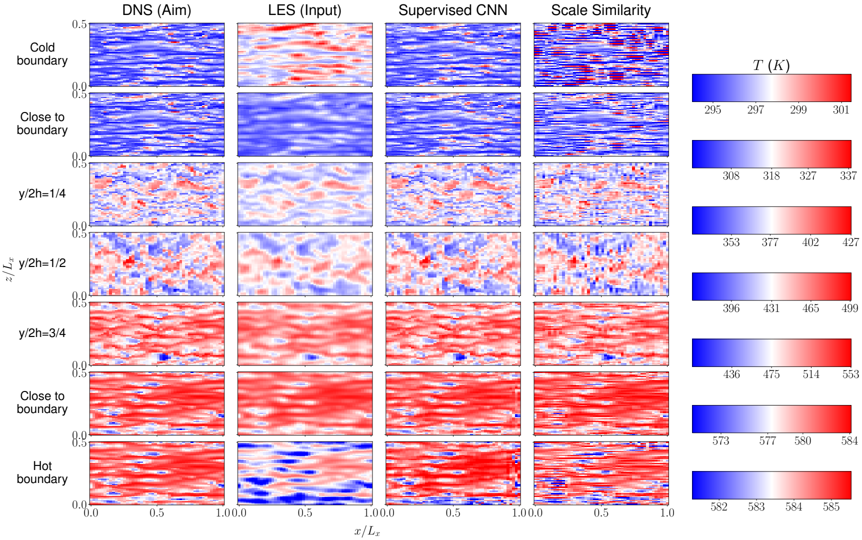
$$\mathcal{L}(T, T_r) = \frac{1}{N_y} \sum_{i=1}^{N_y} (\langle T'^2 \rangle_{x,z,t}(y = y_i) - \langle T_r'^2 \rangle (y = y_i))^2.$$

Algorithms

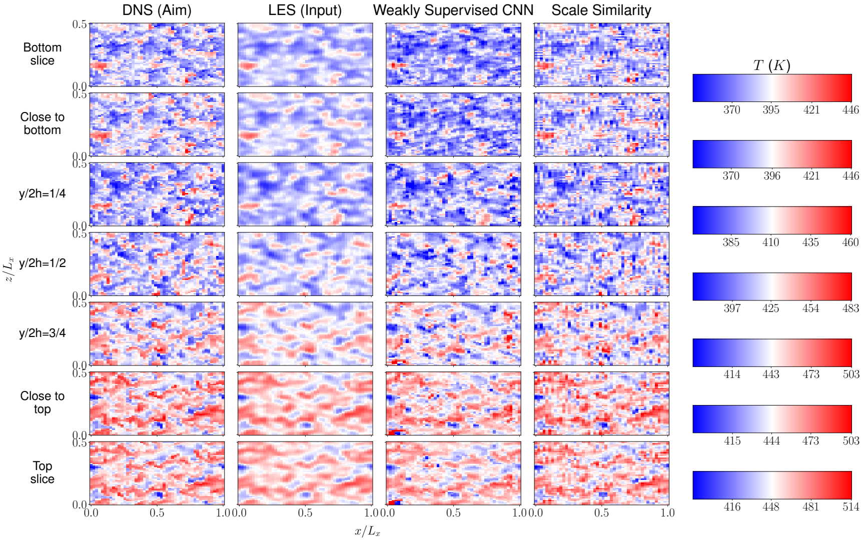
1. **Supervised algorithm** based on the work of Lapeyre *et al.* 2019
   - Architecture: U-Net (1.8M parameters)
   - Loss function: Mean Squared Error over raw data
   - Usable only on *a priori* data
2. **Weakly supervised** algorithm based on the two point correlation
   - Architecture: U-Net (8k parameters)
   - Loss function: Mean Squared Error over two point correlation $\langle T'(x,z)T'(x+X,z+Z)\rangle$
   - Usable on *a priori* and *a posteriori* data
3. **Leaned Van Cittert** algorithm based on the second order temperature statistic
   - Architecture: Single convolution filter (9 parameters)
   - Loss function: Mean Squared Error over second order statistic $\langle T'^2\rangle$
   - Easily usable in both cases

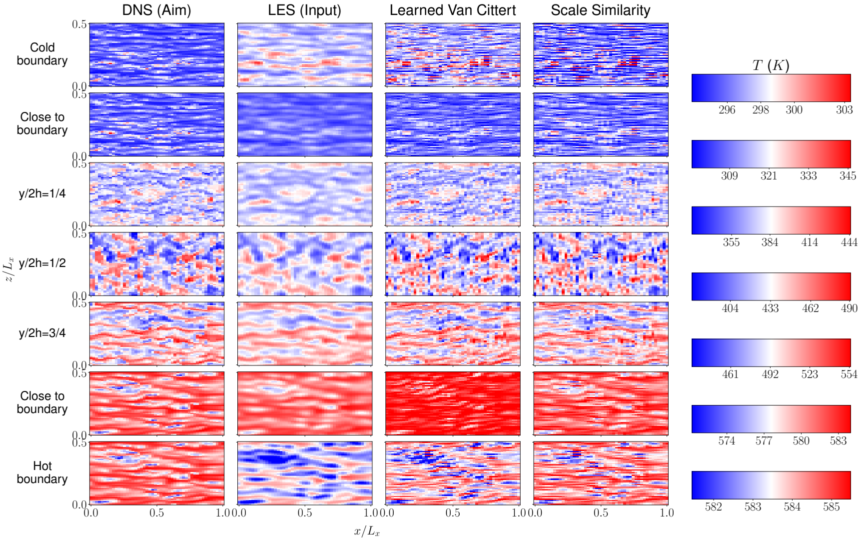
Second order statistic with supervised algorithm and learned Van Cittert method

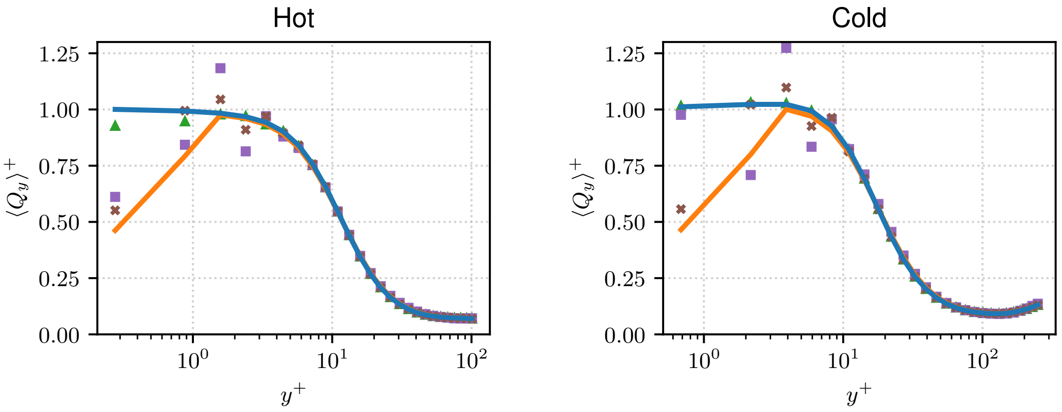




DNS (Aim)
LES (Input)
Supervised CNN
Scale Similarity
Cold boundary
Close to boundary
y/2h=1/4
y/2h=1/2
y/2h=3/4
Close to boundary
Hot boundary
z/L_z
x/L_x
0.0
0.5
1.0
T (K)
295 297 299 301
308 318 327 337
353 377 402 427
396 431 465 499
436 475 514 553
573 577 580 584
582 583 584 585


DNS (Aim)
LES (Input)
Weakly Supervised CNN
Scale Similarity
Bottom slice
Close to bottom
y/2h=1/4
y/2h=1/2
y/2h=3/4
Close to top
Top slice
z/L_x
x/L_x
0.0
0.5
1.0
$T\ (K)$
370 395 421 446
370 396 421 446
385 410 435 460
397 425 454 483
414 443 473 503
415 444 473 503
416 448 481 514

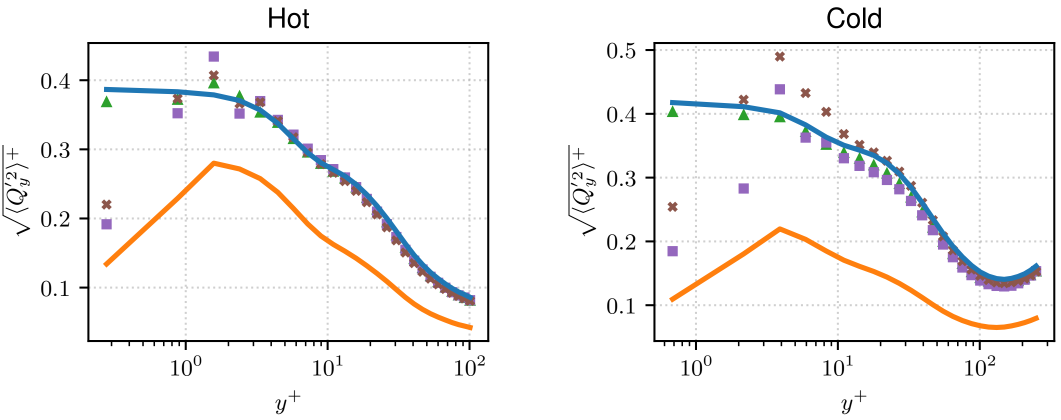

DNS (Aim)
LES (Input)
Learned Van Cittert
Scale Similarity
Cold boundary
Close to boundary
y/2h=1/4
y/2h=1/2
y/2h=3/4
Close to boundary
Hot boundary
z/L_z
x/L_x
0.0
0.5
1.0
$T\ (K)$
296 298 300 303
309 321 333 345
355 384 414 444
404 433 462 490
461 492 523 554
574 577 580 583
582 583 584 585



DNS (Reference) (—) *a priori* LES (Input) (- -)

1. Supervised (▲) 3. Learned Van Cittert (■)

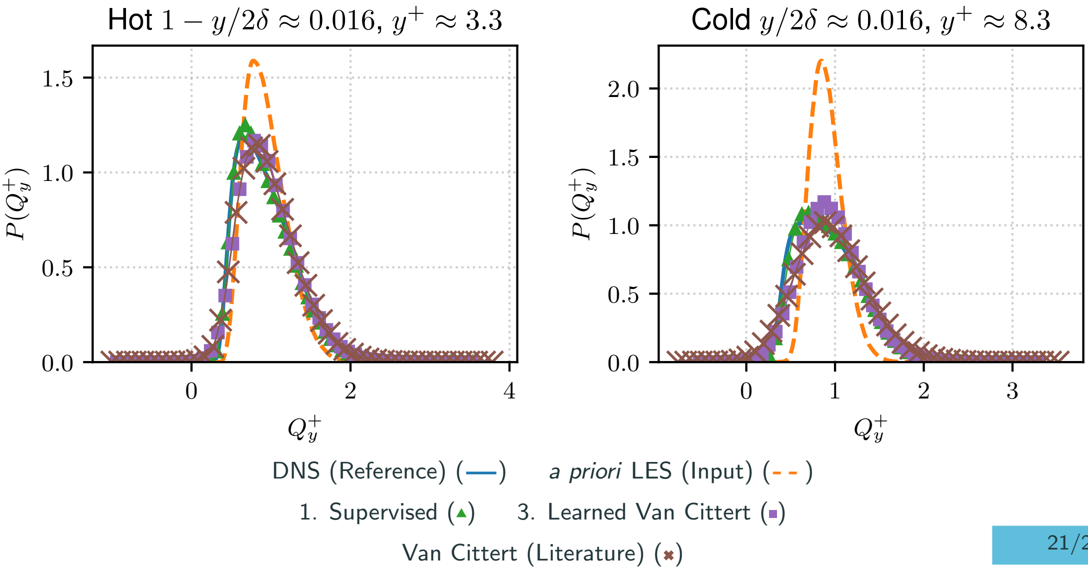
Van Cittert (Literature) (✖)



DNS (Reference) (—) *a priori* LES (Input) (- -)

1. Supervised (▲) 3. Learned Van Cittert (■)

Van Cittert (Literature) (✖)


Hot $1 - y/2\delta \approx 0.016$, $y^+ \approx 3.3$
Cold $y/2\delta \approx 0.016$, $y^+ \approx 8.3$
$P(Q_y^+)$
Q_y^+
0.0
0.5
1.0
1.5
2.0
0
1
2
3
4
DNS (Reference) (—)
a priori LES (Input) (- -)
1. Supervised (▲)
3. Learned Van Cittert (■)
Van Cittert (Literature) (✖)

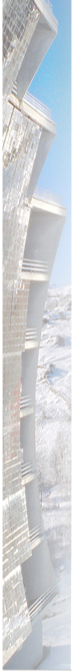
Comparison of Reconstruction Algorithms

- **1. Supervised Algorithm (U-Net):** Best visual and statistical accuracy but limited to *a priori* applications
- **2. Weakly Supervised Algorithm:** Usable *a posteriori*, degraded statistics, visuals and difficult to train
- **3. Learned Van Cittert:** Usable *a posteriori*.
  - ⇒ Minimal training memory
  - ⇒ Physical constraints enforced
  - ⇒ Fits statistics
  - ⇒ Requires improvement w.r.t. padding

Thank you for your attention

Acknowledgements

- **HPC Resources:** GENCI allocations A0132A05099, A0152A14652, and SS012A15404.
- **Software Development:** CEA STMF for the development of the TrioCFD computational software.
- **Funding:** ANR through the ANR-21-CE50-0031 project.



C. J. Lapeyre, A. Misdariis, N. Cazard, D. Veynante, and T. Poinsot. Training convolutional neural networks to estimate turbulent sub-grid scale reaction rates. *Combustion and Flame*, 203:255–264, may 2019. doi: 10.1016/j.combustflame.2019.02.019. URL https://hal.archives-ouvertes.fr/hal-02072920.