The development of an open-source chemical kinetics: VULCAN

Shang-Min Tsai
University of Bern, Switzerland
James Lyons
Arizona State University
Kevin Heng
University of Bern, Switzerland

- open-source Python code
- reduced network
- flexibility

EQ v.s. Kinetics

Equilibrium chemistry	Chemical kinetics
Minimise Gibbs free energy (only final products relevant)	Solve set of mass conservation equations (pathways relevant)
Deals with net reactions only	Deals with transient reactions and radicals
Requires tabulated database of Gibbs free energies of formation	Requires tabulated rate coefficients for both forward and reverse reactions
Does not consider atmospheric circulation or photochemistry	May mimic atmospheric circulation using eddy diffusion

Hierarchical Chemical Network

	Equilibrium thermochemistry	Disequilibrium: vertical mixing	future:
H	started in 2015		
H + O			Disequilibrium: photochemistry & condensation
H + O + C		where we are	
H + O + C + N		by the end of 2015	

chemical complexity

Model Description

- Reduced network (25 species & ~300 reactions)
  (H,H2,O,OH,H2O,CH,C,CH2,CH3,CH4,C2,C2H2,C2H3,C2H,C2H4,C2H5,
  C2H6,C4H2,CO,CO2,CH2OH,H2CO,HCO,CH3O,CH3OH)
- Group mechanism
- Rate coefficient from NIST / reversed
- solving stiff ODEs: $\frac{dY}{dt} = f(Y,t) = P - L$
- semi-implicit Euler solver(analytical Jacobian matrix)

$$f(Y^{n+1},t^{n+1}) \approx f(Y^n,t^n) + \left.\frac{\partial f}{\partial Y}\right|_{Y^n} \cdot (Y^{n+1} - Y^n)$$

- Crank-Nicolson method for diffusion
- Challenge: stiffness & coding time

Auto-mated generated script

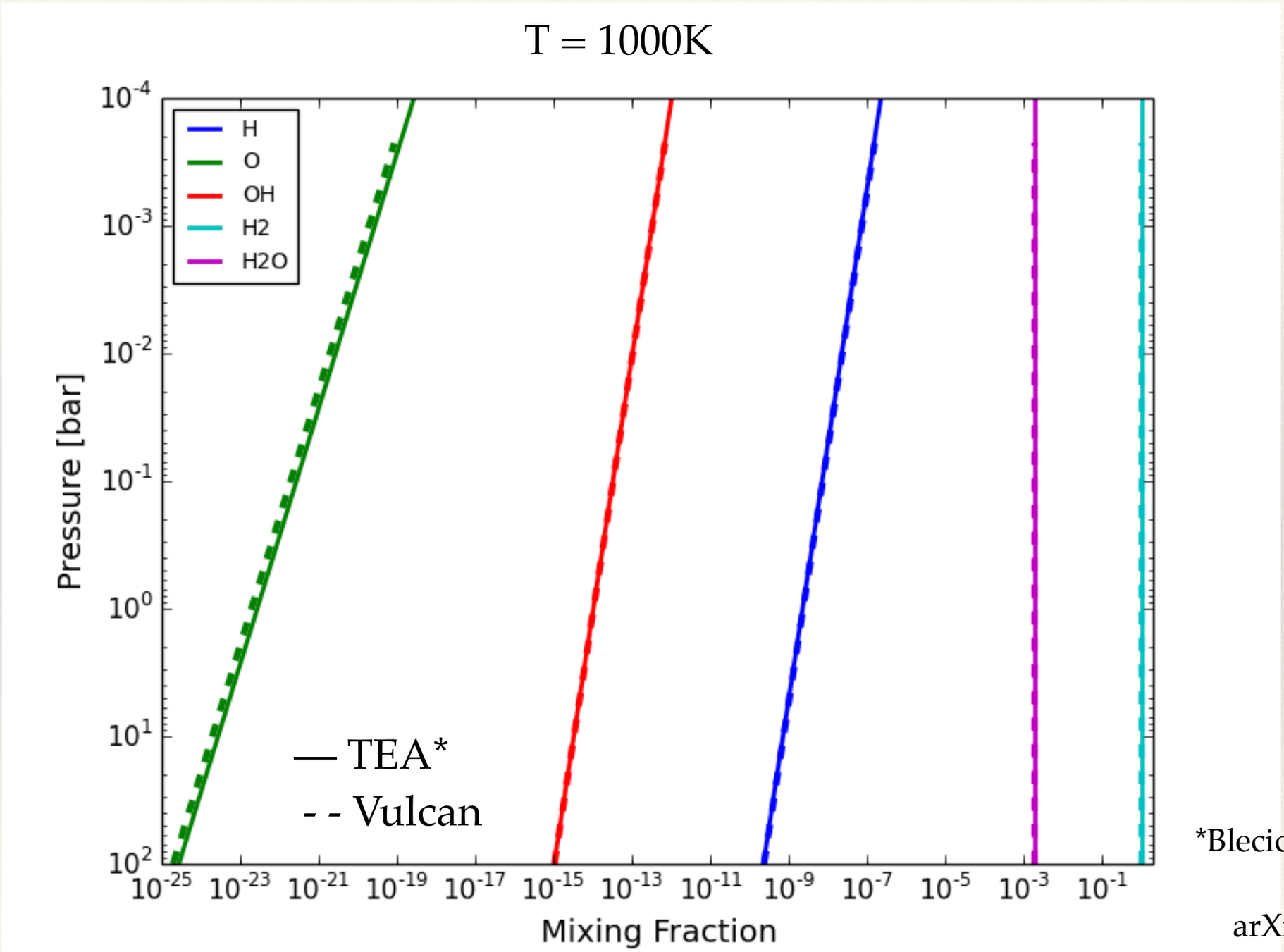
```
# C/H/O chemistry

#R1
H + H + M -> H2 + M
#R3
H + O + M -> OH + M
#R5
H + H2O -> OH + H2
#R7
O + H2 -> OH + H
#R9
O + H2O -> 2*OH
#R11
OH + H + M -> H2O + M
#R13
H + CH -> H2 + C
#R15
H + CH + M -> CH2 + M
#R17
H + CH2 -> CH + H2
#R19
CH2 + H2 -> H + CH3
#R21
H + CH3 + M -> CH4 + M
#R23
H + CH4 -> CH3 + H2
#R25
C + CH -> C2 + H
#R27
CH3 + C -> H + C2H2
```

- flexibility of the model

inspired from Kinpy

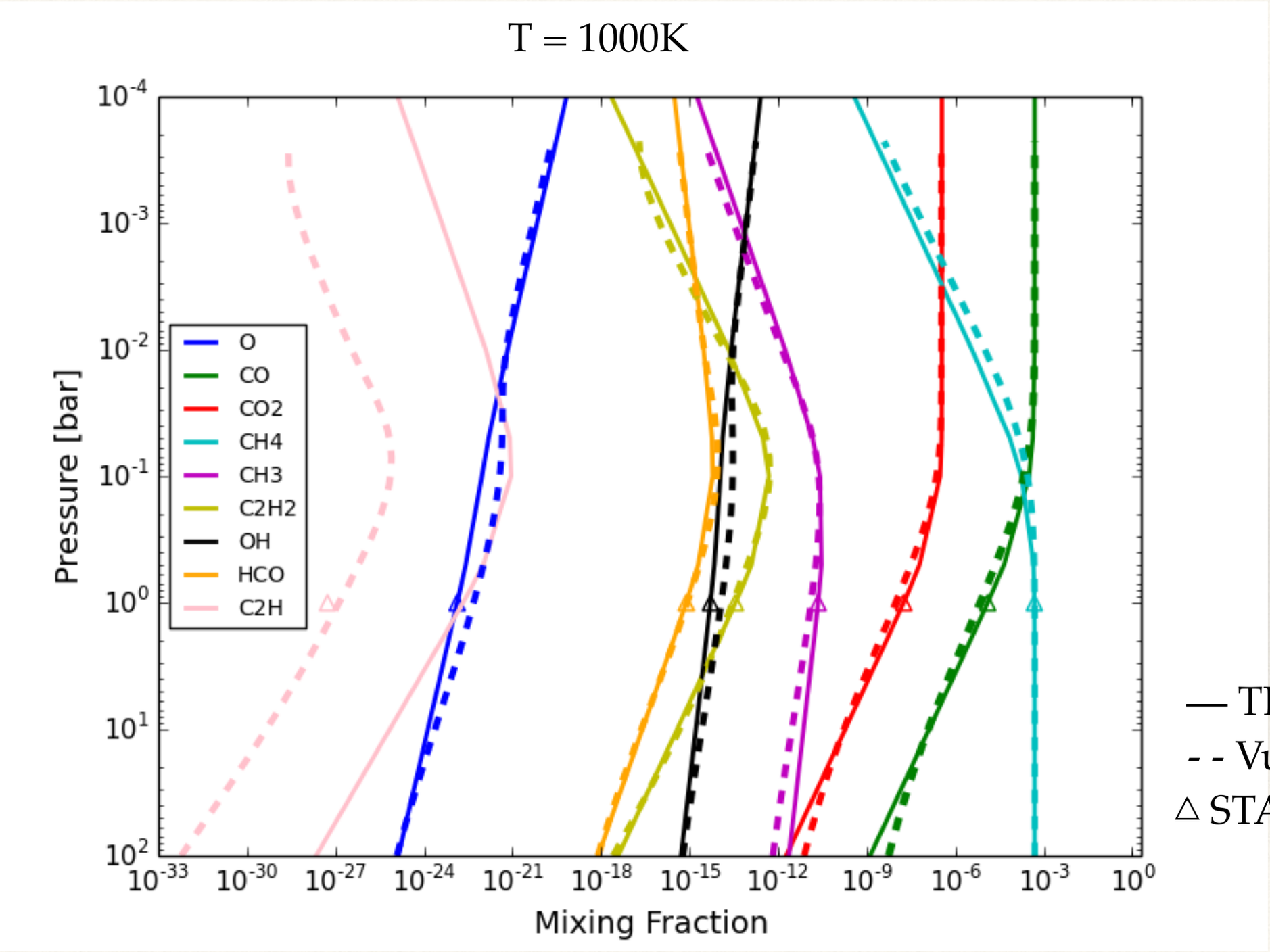
```
## Mapping ##

H: y[0], H2: y[1], O: y[2], OH: y[3], H2O: y[4], CH: y[5], C: y[6],

H   0    -1*v_1(y[1], y[0]) -1*v_1(y[1], y[0]) -1*v_3(y[0], y[2], y
H2  1    +1*v_1(y[1], y[0]) +1*v_5(y[1], y[0], y[4], y[3]) -1*v_7(y
O   2    -1*v_3(y[0], y[2], y[3]) -1*v_7(y[1], y[0], y[2], y[3]) -1
OH  3    +1*v_3(y[0], y[2], y[3]) +1*v_5(y[1], y[0], y[4], y[3]) +1
H2O 4    -1*v_5(y[1], y[0], y[4], y[3]) -1*v_9(y[4], y[2], y[3]) +1
CH  5    -1*v_13(y[1], y[0], y[6], y[5]) -1*v_15(y[0], y[7], y[5])
C   6    +1*v_13(y[1], y[0], y[6], y[5]) -1*v_25(y[10], y[6], y[5],
CH2 7    +1*v_15(y[0], y[7], y[5]) -1*v_17(y[1], y[0], y[7], y[5])
CH3 8    +1*v_19(y[1], y[0], y[7], y[8]) -1*v_21(y[0], y[8], y[9])
CH4 9    +1*v_21(y[0], y[8], y[9]) -1*v_23(y[1], y[0], y[8], y[9])
C2  10   +1*v_25(y[10], y[6], y[5], y[0]) +1*v_65(y[10], y[6]) -1*v
C2H2    11   +1*v_27(y[0], y[6], y[8], y[11]) -1*v_29(y[0], y[12],
C2H3    12   +1*v_29(y[0], y[12], y[11]) -1*v_35(y[1], y[0], y[12],
C2H 13   -1*v_31(y[1], y[0], y[13], y[11]) -1*v_55(y[0], y[13], y[1
C2H4    14   +1*v_37(y[0], y[14], y[12]) +1*v_39(y[1], y[0], y[14],
C2H5    15   +1*v_45(y[0], y[15], y[14]) -1*v_47(y[0], y[15], y[8])
C2H6    16   +1*v_51(y[0], y[15], y[16]) -1*v_53(y[1], y[0], y[15],
C4H2    17   +1*v_55(y[0], y[13], y[17], y[11])
CO  18   -1*v_57(y[0], y[19], y[18], y[3]) +1*v_63(y[1], y[4], y[19
CO2 19   +1*v_57(y[0], y[19], y[18], y[3]) -1*v_63(y[1], y[4], y[19
CH2OH   20   +1*v_167(y[20], y[8], y[0], y[3]) -1*v_169(y[20], y[21
H2CO    21   +1*v_169(y[20], y[21], y[0]) -1*v_171(y[1], y[0], y[21
HCO 22   +1*v_171(y[1], y[0], y[21], y[22]) -1*v_173(y[18], y[0], y
CH3O    23   +1*v_195(y[0], y[23], y[21]) -1*v_197(y[23], y[21], y[
CH3OH   24   +1*v_201(y[24], y[20], y[0]) +1*v_203(y[24], y[8], y[3
CH3CO   25   +1*v_223(y[25], y[11], y[3]) -1*v_267(y[8], y[25], y[1
'''


def chemdf(y):
    y = np.transpose(y)
    dydt = np.zeros(shape=y.shape)
    dydt[0] =  -1*v_1(y[1], y[0]) -1*v_1(y[1], y[0]) -1*v_3(y[0], y
    dydt[1] =  +1*v_1(y[1], y[0]) +1*v_5(y[1], y[0], y[4], y[3]) -1
    dydt[2] =  -1*v_3(y[0], y[2], y[3]) -1*v_7(y[1], y[0], y[2], y[
    dydt[3] =  +1*v_3(y[0], y[2], y[3]) +1*v_5(y[1], y[0], y[4], y[
    dydt[4] =  -1*v_5(y[1], y[0], y[4], y[3]) -1*v_9(y[4], y[2], y[
```

Reproduce Chemical EQ



*Blecic, J., Harrington, J., & Bowman arXiv:1505.06392v1

Reproduce Chemical EQ



Transport-induced quench

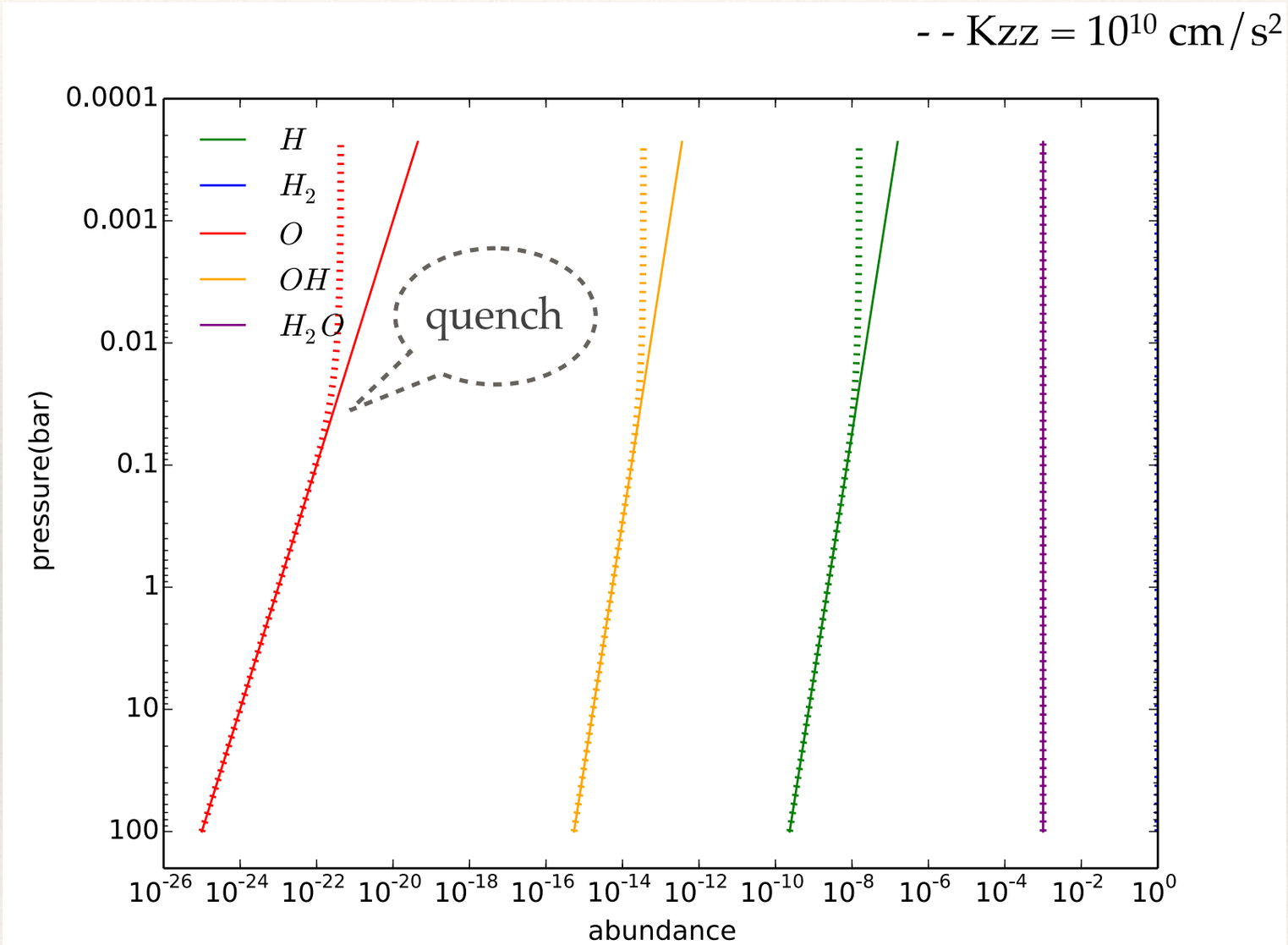
- $\tau_{mix} = L^2/K_{zz}$

  $\tau_{chem} = [x]/\ (d[x]/dt)$

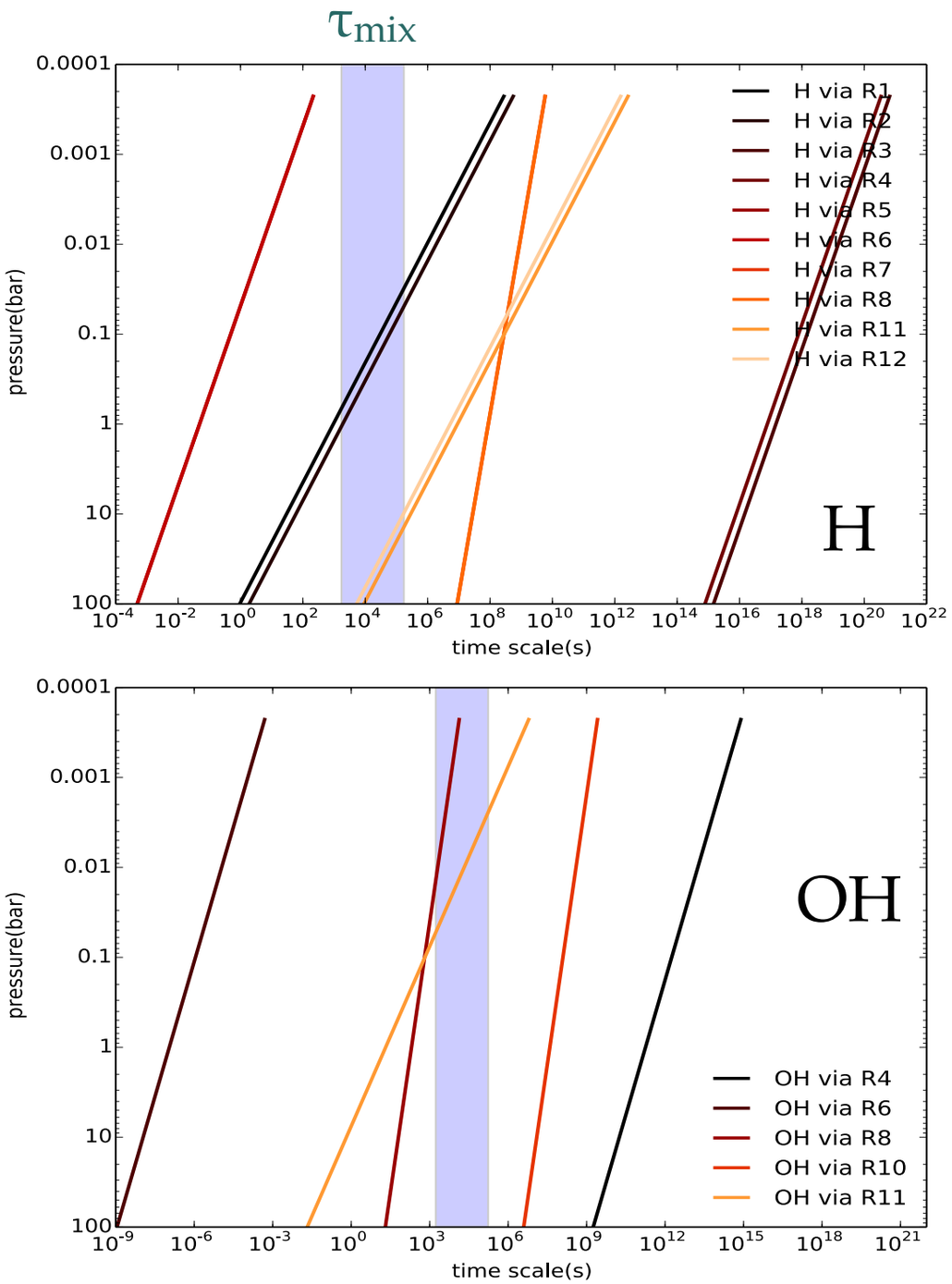
- quench level:

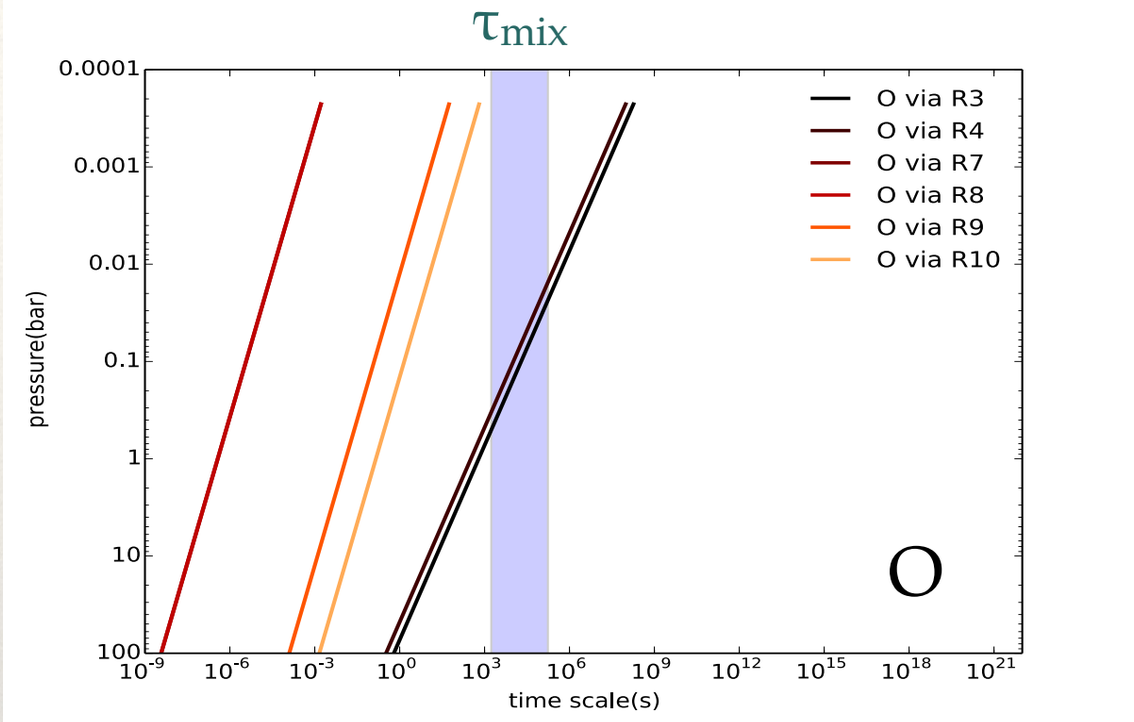
  $\tau_{chem} = \tau_{mix}$

- Identifying the rate-limiting step is crucial!



Transport-induced quench



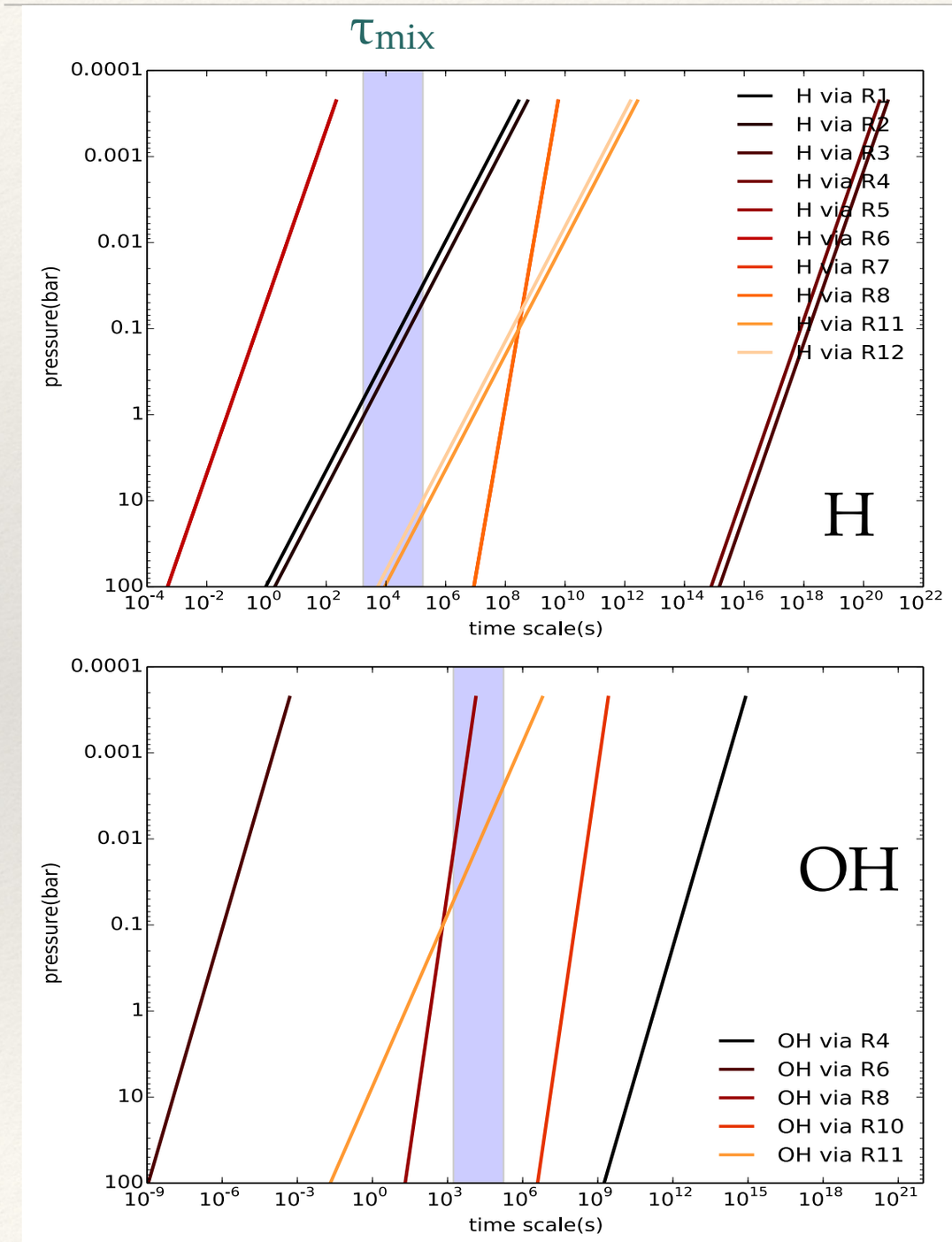


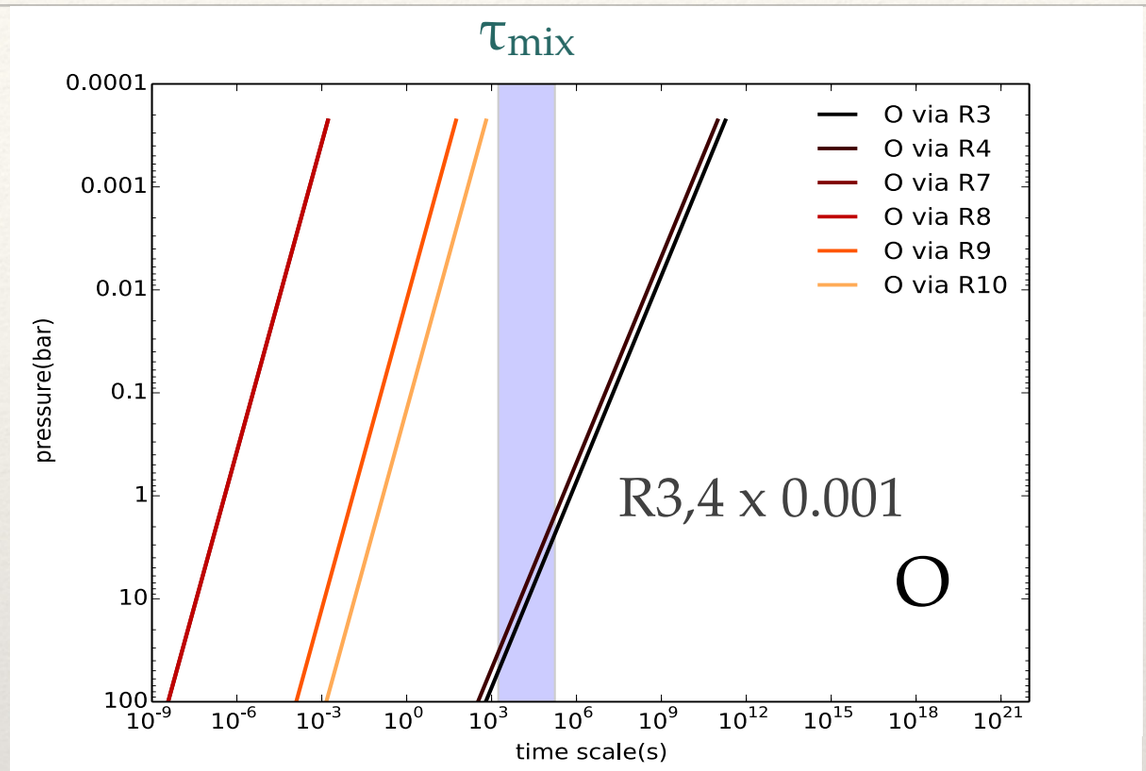
R1,2: H + H + M <-> H2 + M

R3,4: H + O + M <-> OH + M

R11,12: OH + H + M -> H2O + M

Transport-induced quench



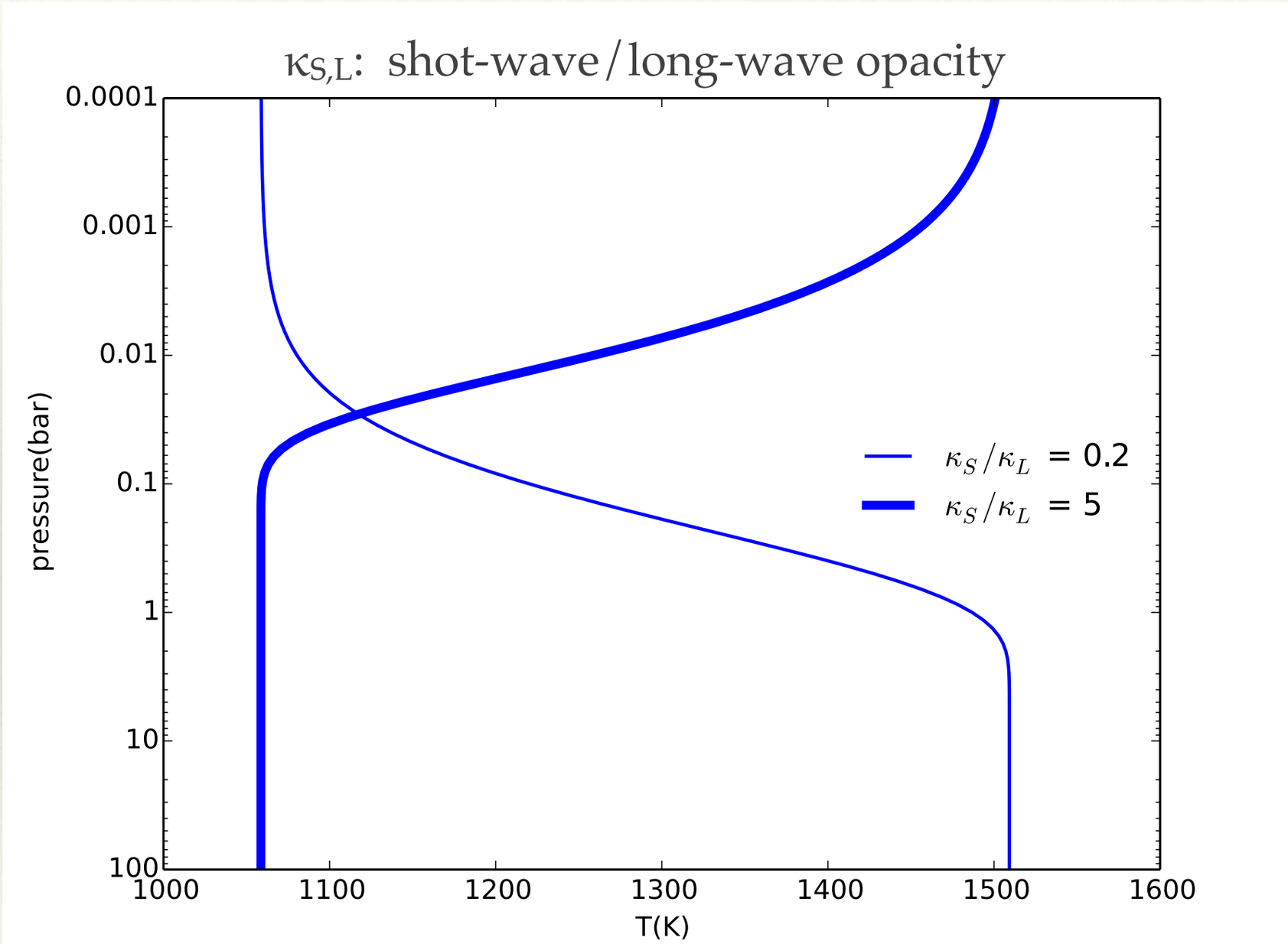


R1,2: H + H + M <-> H2 + M

R3,4: H + O + M <-> OH + M

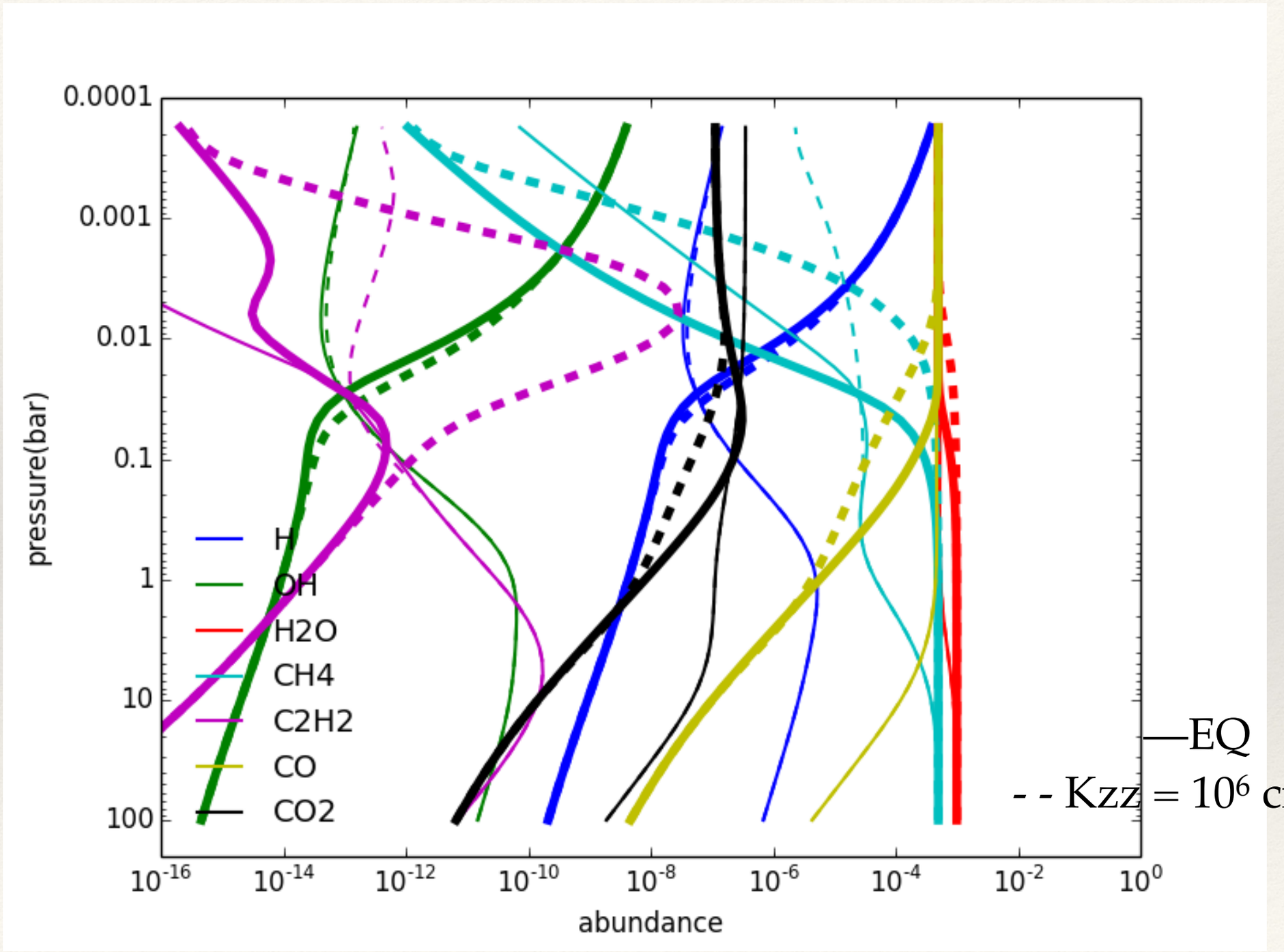
R11,12: OH + H + M -> H2O + M

Test on TP-profile



Heng K., Mendonca J. M., Lee J.-M., 2014, ApJS

Test on TP-profile



—EQ

- - Kzz = 10^6 cm/s^2

ESP

EXOCLIMES SIMULATION PLATFORM

Our dream: an open-source set of modern computer codes, established as community tools

Luc Grosheintz, Matej Malik, Baptiste Lavie,
Shang-Min Tsai, Maria Oreshenko,
Joao Mendonca, Simon Grimm,
Daniel Kitzmann, Frank Wagner
Collaborators: Sid Mishra, Roger Kaeppeli,
Jim Lyons, Adam Burrows, Mike Line

Professor Kevin Heng
University of Bern, Switzerland

- The Exoclimes Simulation Platform (ESP) is a dream and vision to provide a versatile set of codes, for understanding exoplanetary atmospheres, to the scientific community.

- The ESP has three main components: HELIOS (radiation), VULCAN (chemistry) and THOR (fluid dynamics).