Hybrid CUDA/JAVA model for GPU Calculations of Gas Line-by-Line Absorption of Atmospheric Radiation

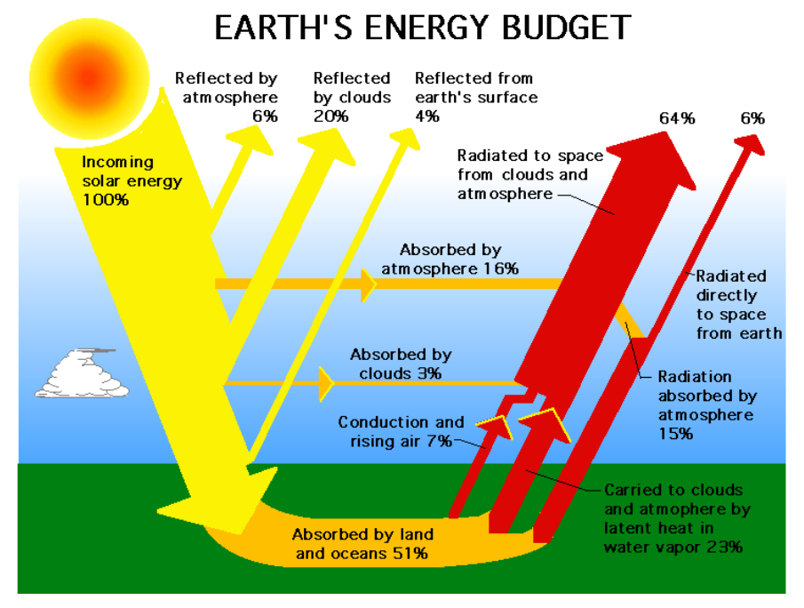
William F. Godoy

NASA Langley Research Center, Hampton, Virginia

Abstract

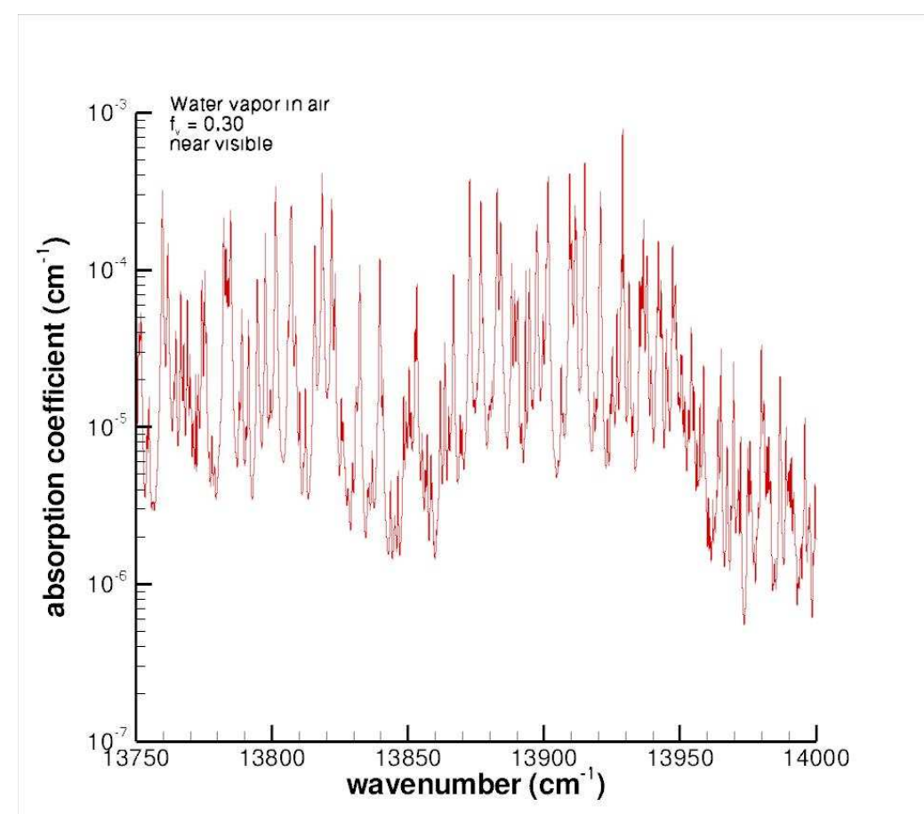
This work shows the potential in the use of graphics processing units (GPU) to speed up the calculation of radiative energy absorption by atmospheric gases. Gas absorption calculations are needed at millions of spectral lines (wavelengths) in order to have an accurate depiction of the Earth's in-coming and out-coming radiative energies. The CUDA/GPU portion of the code obtains fast Voigt lineshapes at several spectral locations, whereas the NIO/Java portion of the code performs efficient random file access on the large HITRAN database to obtain molecular gas parameters. The first advantage of this programming model is the great speed up obtain by the introduction of parallel GPU algorithms, and the second advantage is the modular combination of the lower-level CUDA algorithms for the GPU calculations and the higher-level Java language for efficient input/output of files (I/O). The latter results in an accessible interface to the end-user that is not an expert in GPU programming.

Introduction



- Modeling radiative transfer in the atmosphere is an extremely challenging task
- Current needs :
  - Fast, accurate and robust PARALLEL models (GPU, MPI, Threads)
  - Sub-grid models for coupling with other phenomena (*i.e.* turbulence, convection, cloud physics)
  - Avoid computationally expensive codes (*i.e.* Monte Carlo)
- General computing in graphic processing (GPGPU) units is a powerful alternative
- Drawbacks:
  - Full speed up only for single precision (4 bytes)
  - GPU low memory and CPU/GPU communication are bottlenecks
  - Lack of documentation, validation and applications to radiative transfer

Mathematical Formulation



- Gas absorption is a extremely noisy function along the spectrum [1]
- Line-by-line (LBL) calculations using Voigt lineshapes are required for every single gas at every single state [2]
- Slow process as there is many gas species and thousands of millions of important wavenumbers

CUDA GPU / JAVA IO Programming Model

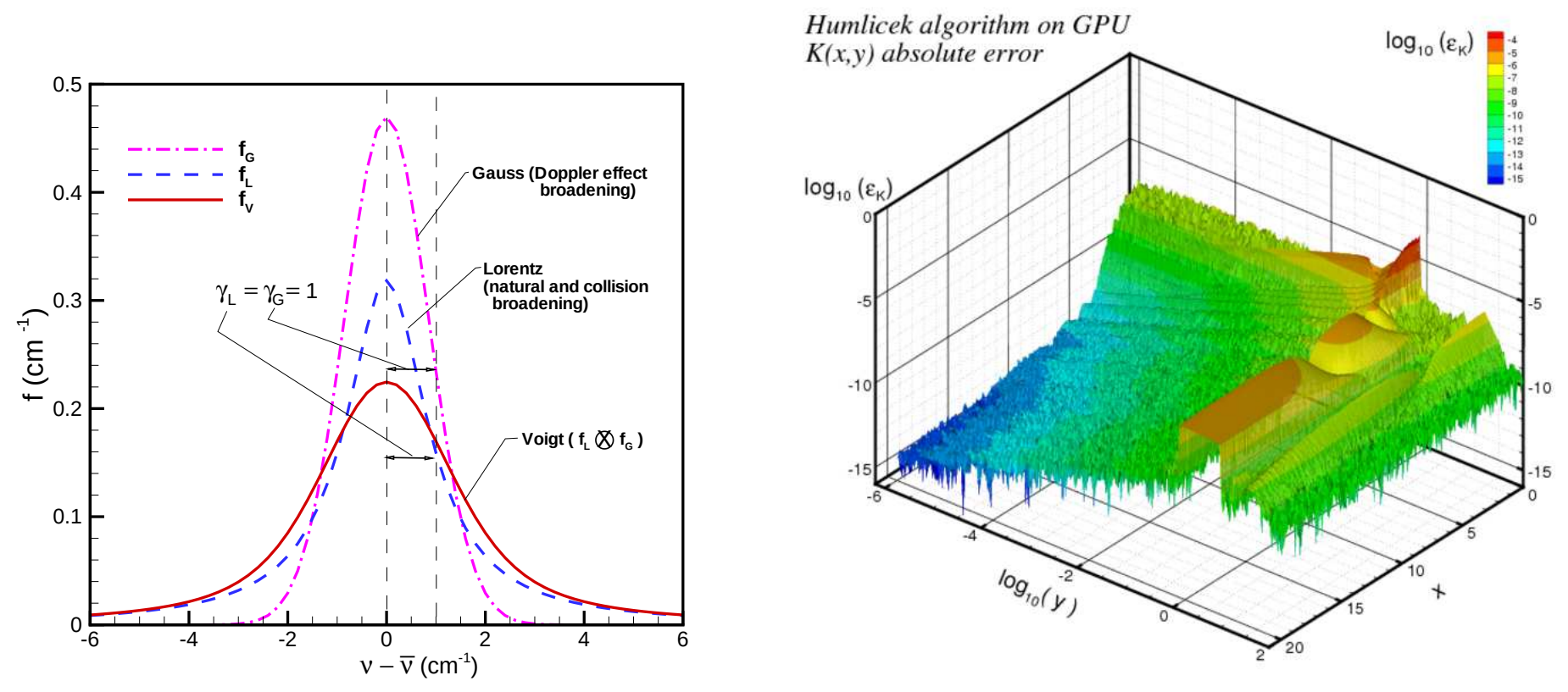
GPU Parallel Model: Voigt Lineshape - Integral Function[2]

$$f_{V\,(\nu-\bar{\nu},\gamma_L,\gamma_G)} = \int_{-\infty}^{\infty} f_{L\,(\nu-\nu',T,p)}\, f_{G\,(\nu'-\bar{\nu},T)} d\nu \qquad (1)$$

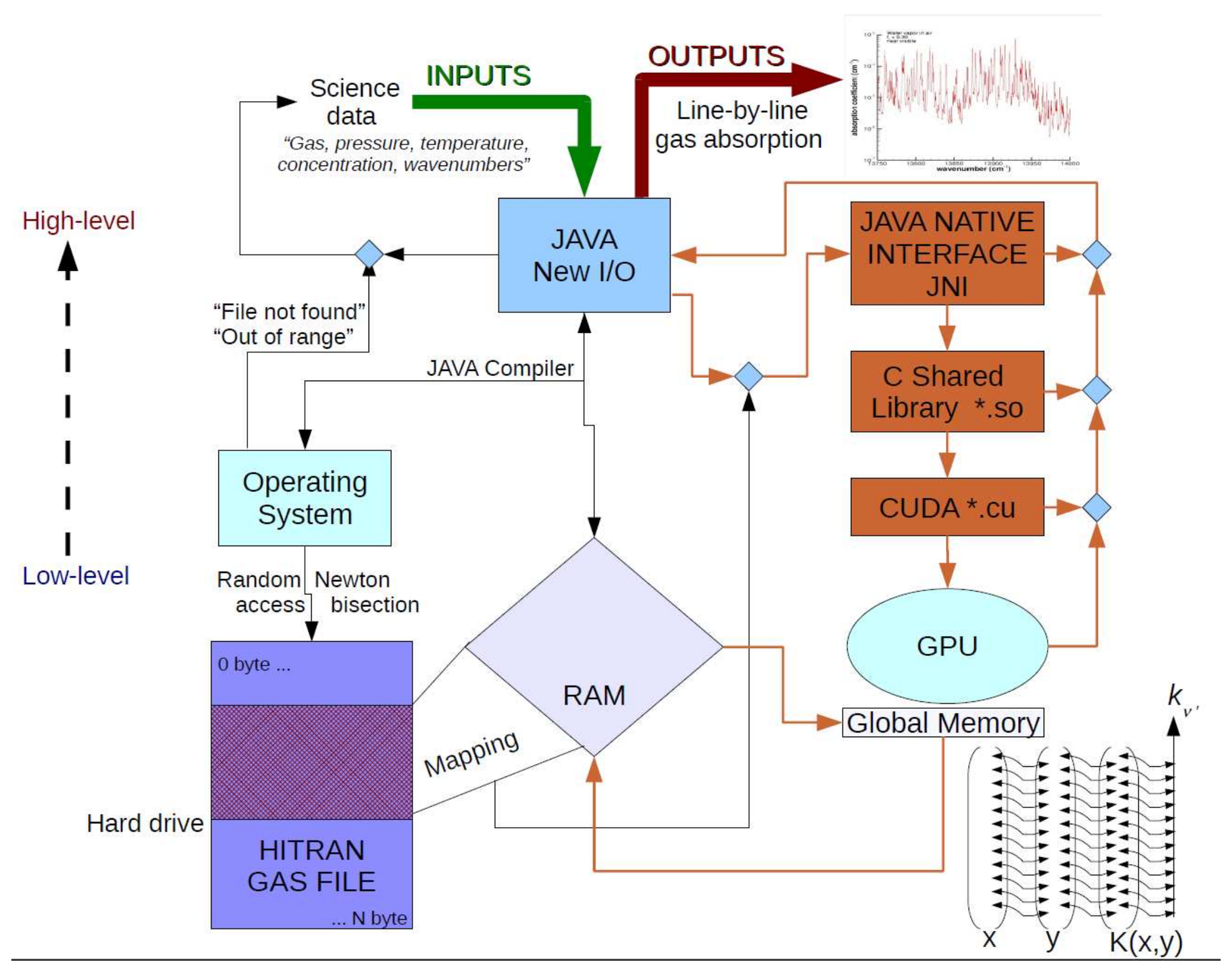
$$= \frac{\sqrt{\ln 2}}{\sqrt{\pi}\,\gamma_G} K(x,y) = \mathrm{Re}\left[e^{-z^2}\left(1 - erf[-\mathrm{i}z]\right)\right] \qquad (2)$$

$$x = \sqrt{\ln 2}\,\frac{\nu-\bar{\nu}}{\gamma_G} \qquad y = \sqrt{\ln 2}\,\frac{\gamma_L}{\gamma_G}. \qquad (3)$$

Voigt lineshape calculation on GPU

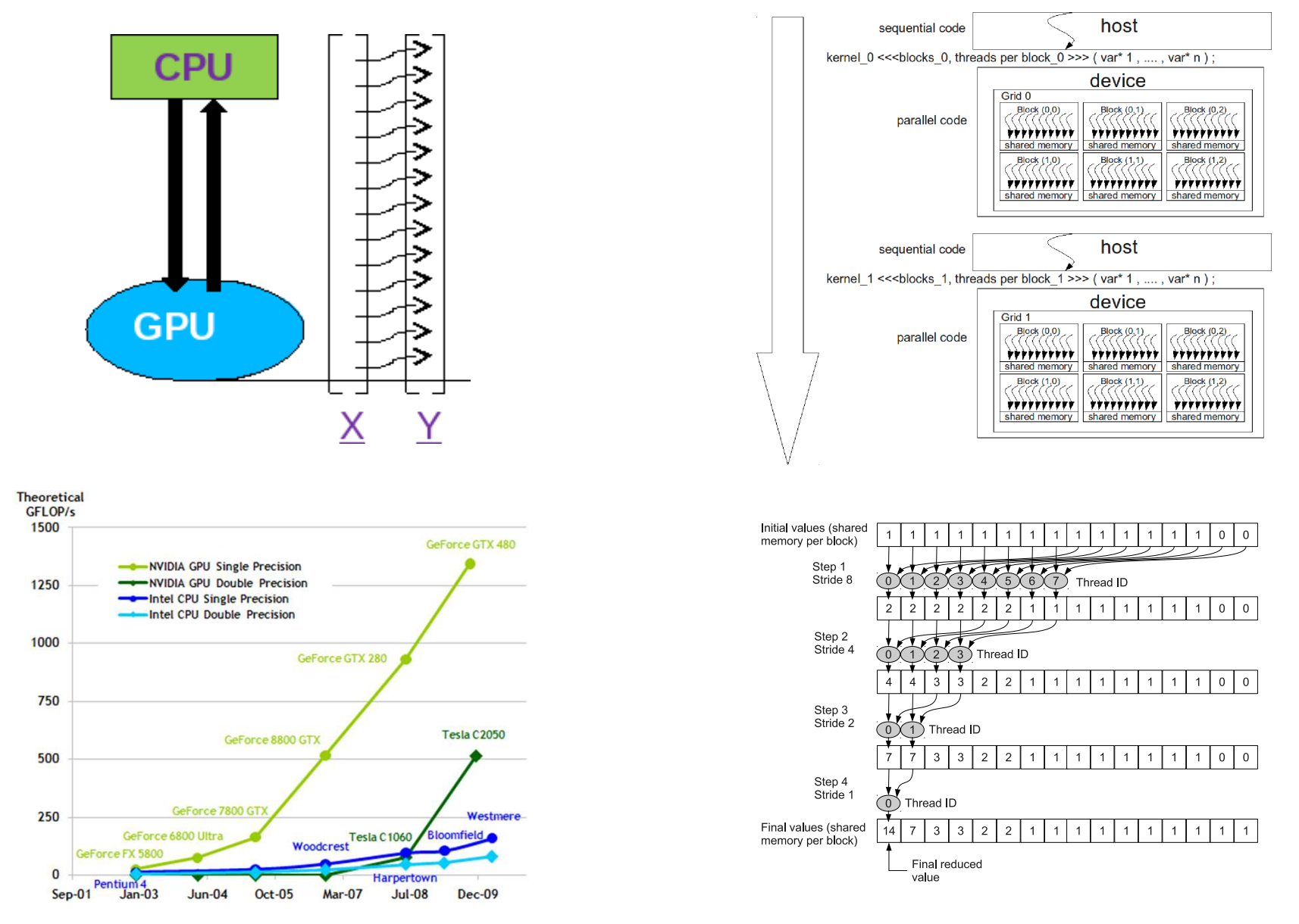


CUDA / JAVA flow diagram



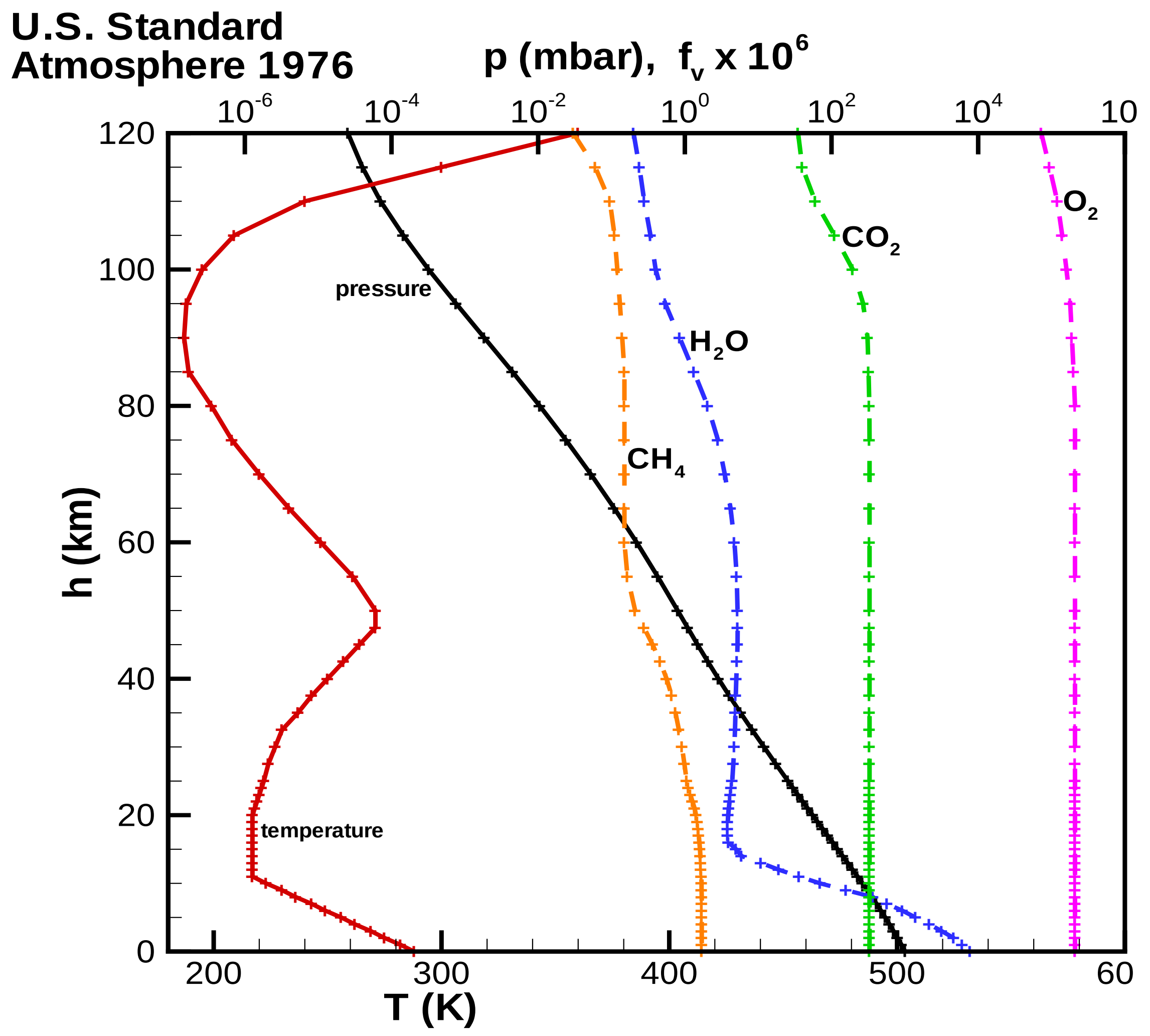
GPGPU programming basis [3, 4]

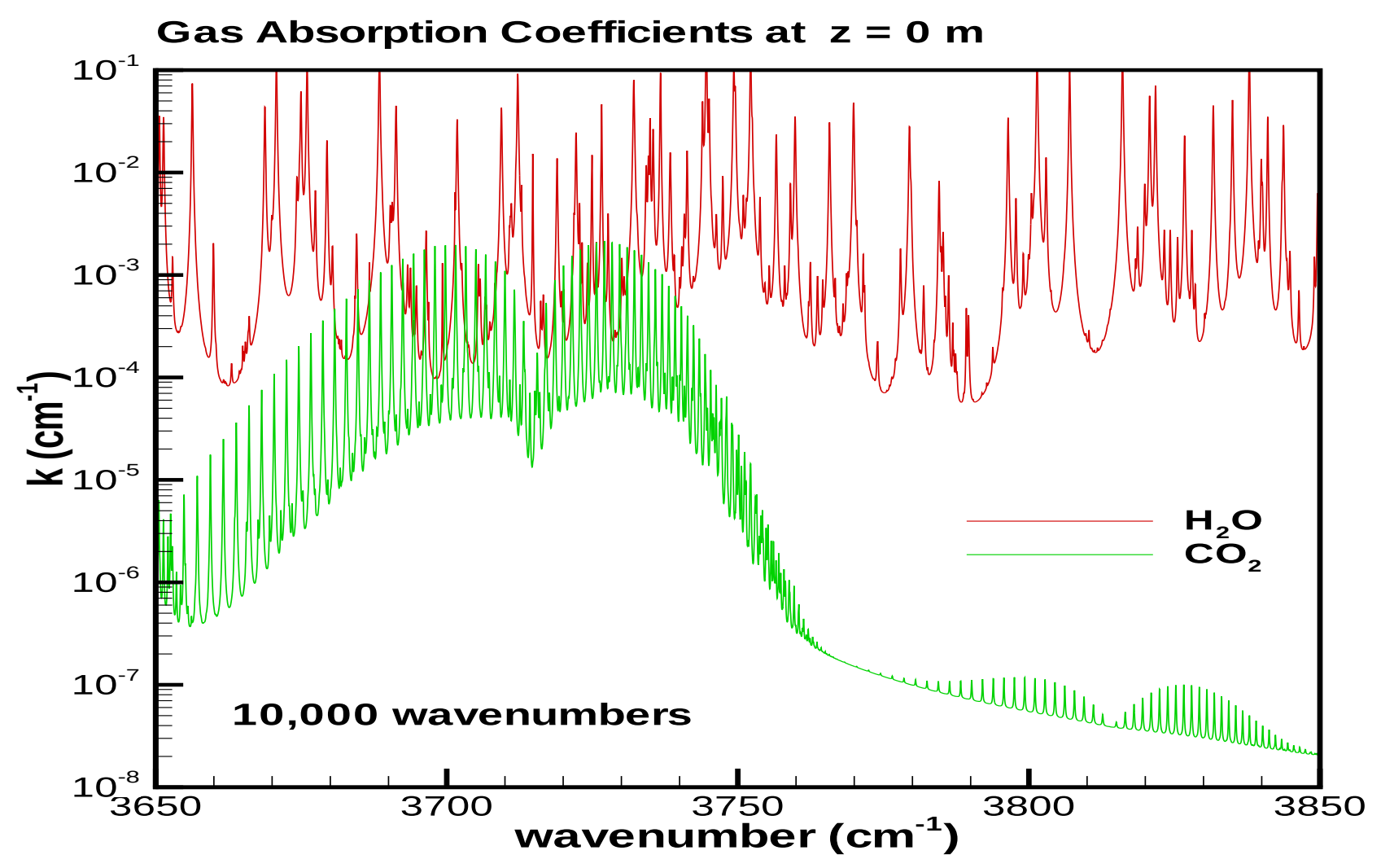
- Cost, \$ per teraflop (10^{12} floating point operations) is low
- A GPU has thousands of processors for lightweigth tasks
- Parallel processes (threads) launch from CPU
- High CPU/GPU data transfer reduces efficiency
- Double precision is needed for Jacobian-free calculations

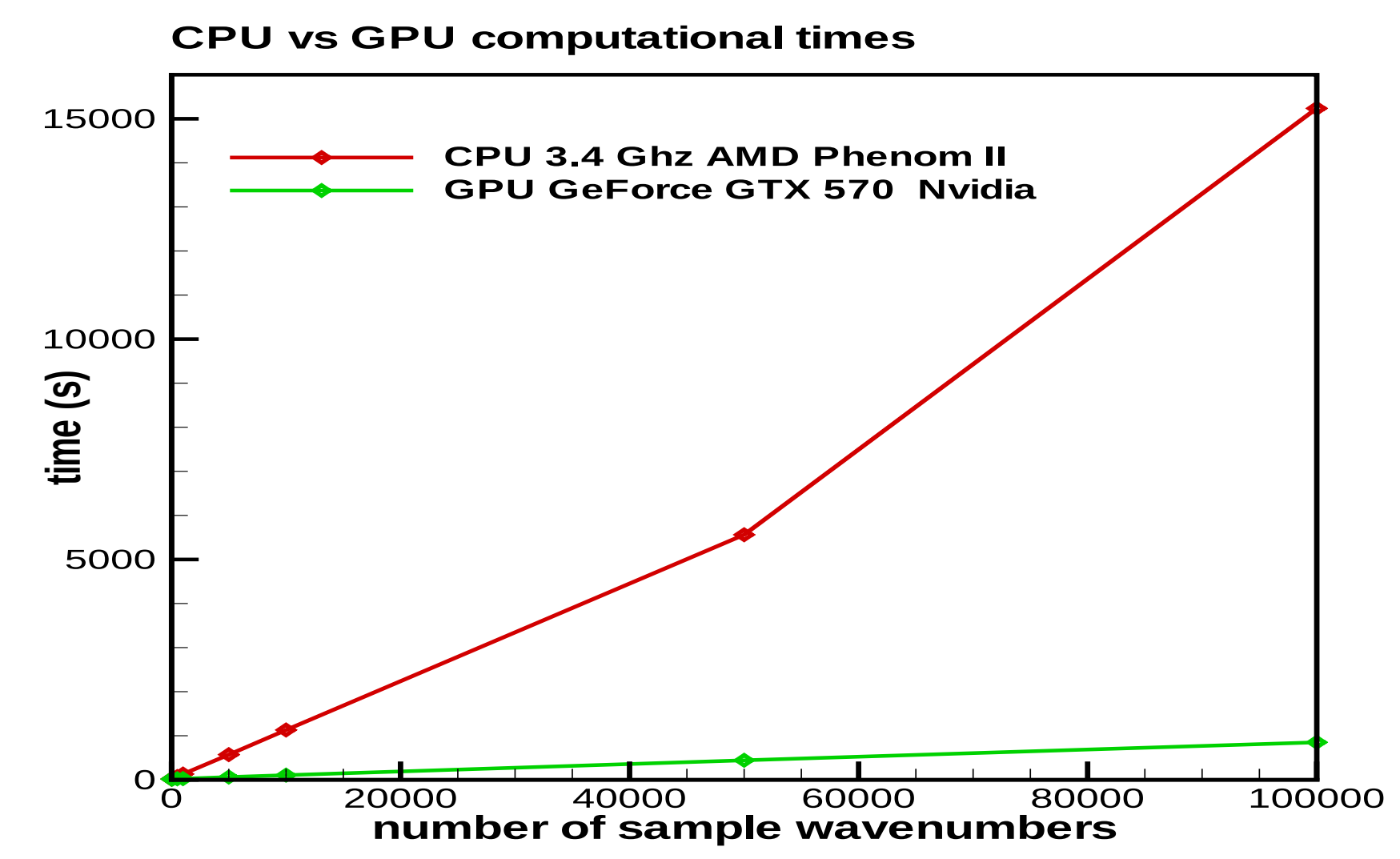


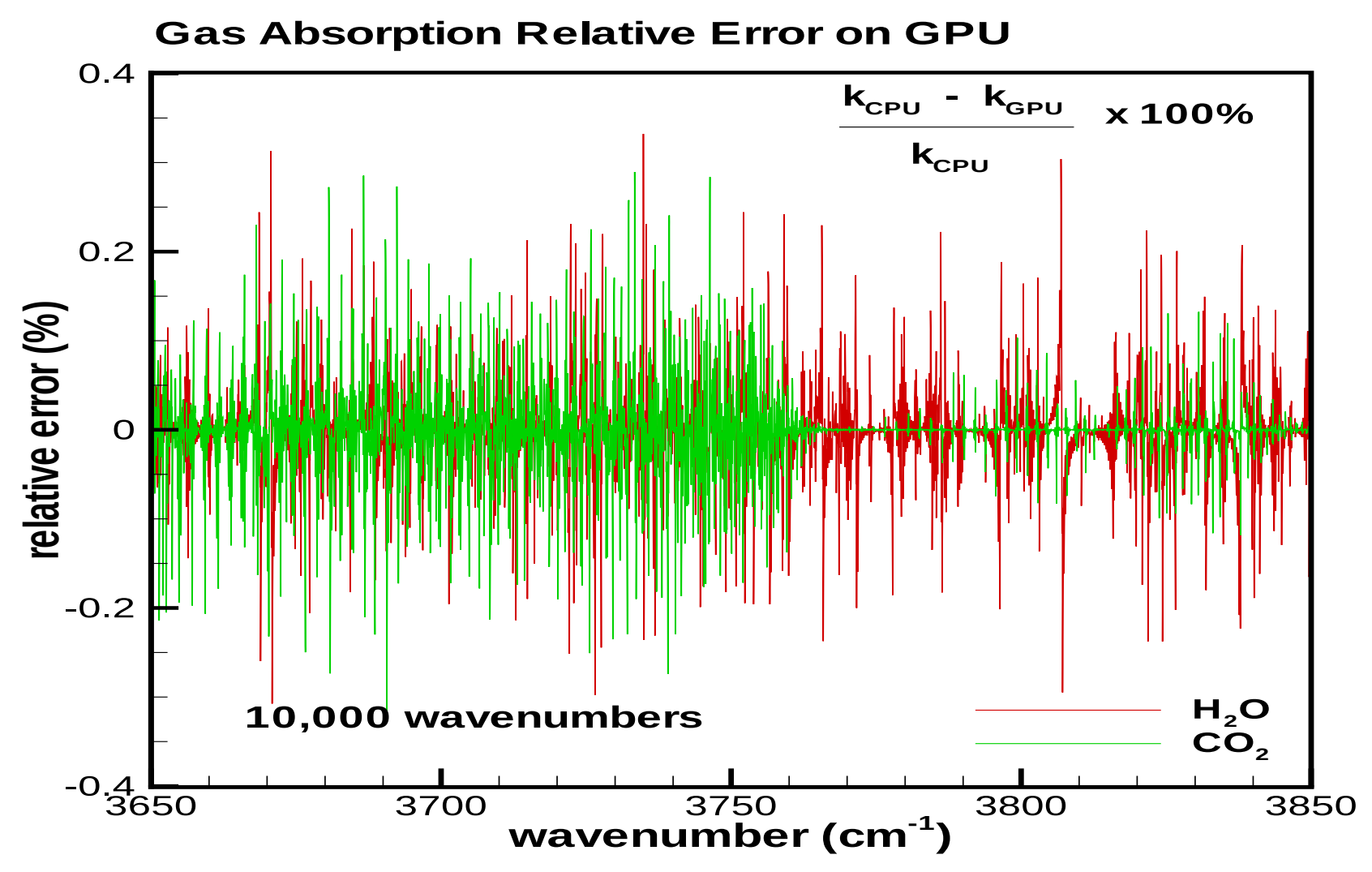
Results and Speed up for Atmospheric H_2O and CO_2

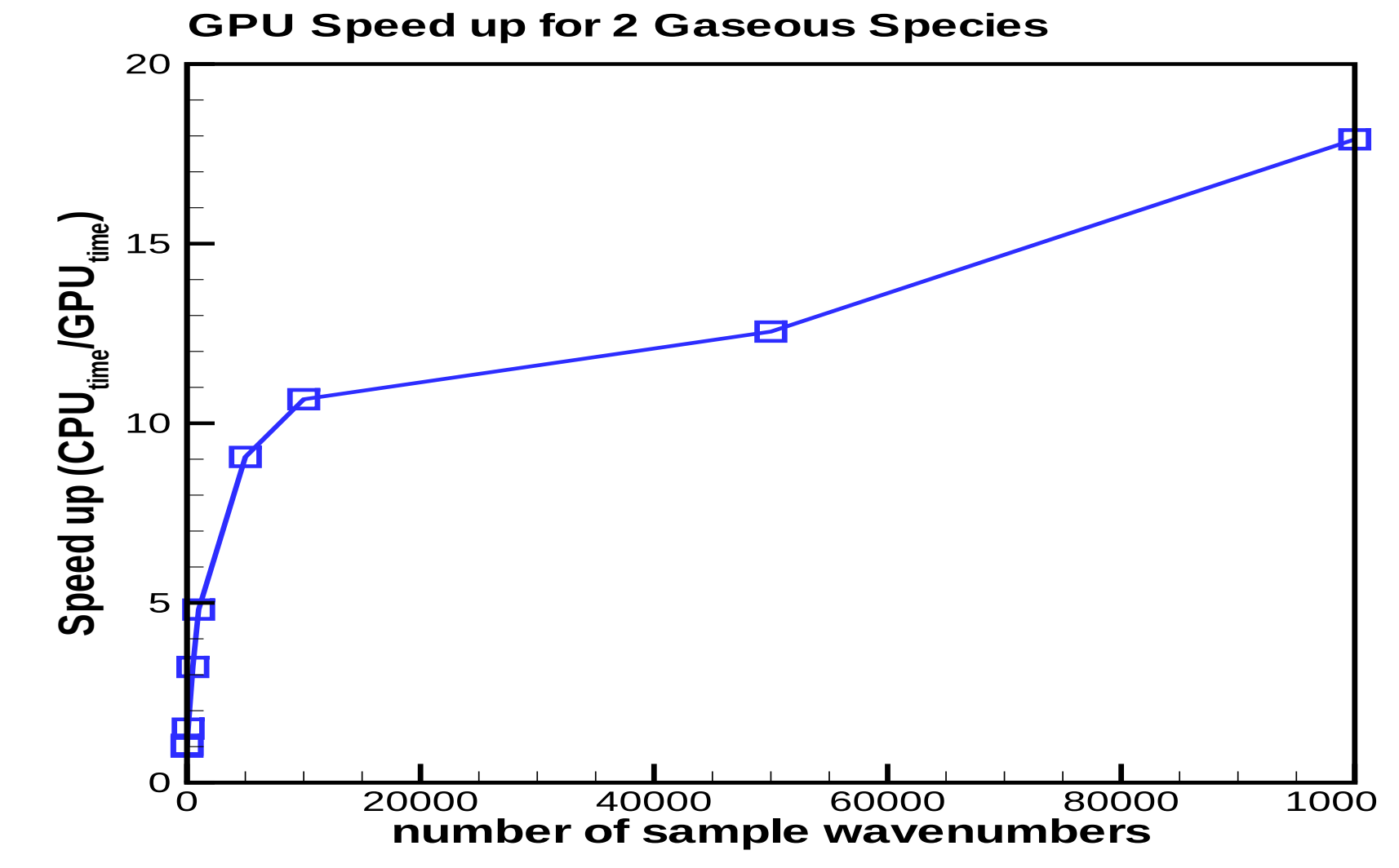
Atmospheric Profile at 50 Altitudes











Summary & Conclusions

- A hybrid Java/CUDA model allows the modular combination of GPU and high-level I/O algorithms
- The model allows for code reusability with an accesible learning curve for the end user
- Errors due to GPU single precision are less than 0.3% in the case of atmospheric H_2O and CO_2 at sea level
- Maximum speed up of 18 for 50,000 tested wavenumbers, speed ups grow proportional to the number of wavenumbers
- Potential extension to entire spectrum and multiple gaseous species is a must!

References

[1] R. M. Goody, Y. L. Yung, Atmospheric Radiation: Theoretical Basis, 2nd Edition, Oxford University Press, New York, NY, 1995.

[2] M. F. Modest, Radiative Heat Transfer, 2nd Edition, Academic Press, Elsevier Science, San Diego, CA, 2003.

[3] CUDA: Compute Unified Device Architecture, NVIDIA Corporation., http://www.nvidia.com/object/cuda_home_new.html, 2010.

[4] W. F. Godoy, X. Liu, Introduction of parallel GPGPU acceleration algorithms for the solution of radiative transfer, Num Heat Trans, Part B: Fundamentals 59 (2011) 1–25.