Fast GPU Fitting of Kinetic Models for Dynamic PET

Benjamin R. Smith Ghassan Hamarneh Ahmed Saad

MICCAI 2010 – Workshop on High-Performance Medical Image Computing for Image-Assisted Clinical Intervention and Decision Making

September 24th, 2010

Presented by: Ahmed Saad

Contact: brsmith@cs.sfu.ca

Overview

Introduction
- Dynamic PET
- Kinetic Modelling

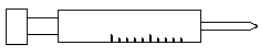
Motivation
- Challenges of Fitting Kinetic Models
- Proposed Solution

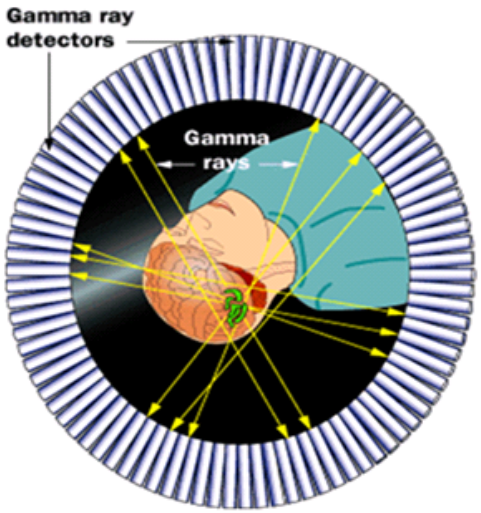
Method
- Analytic Kinetic Fitting
- Parallel Levenberg-Marquardt Algorithm

Results

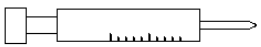
Conclusion

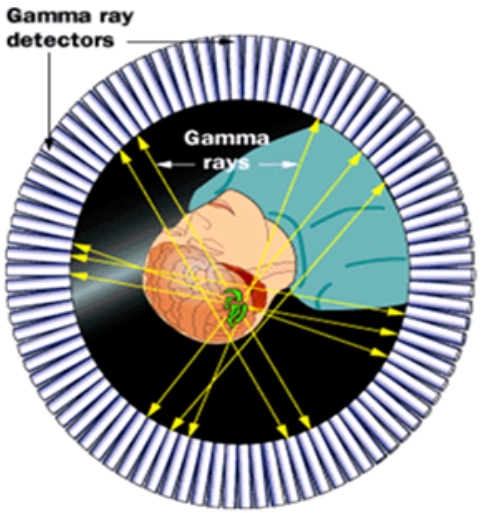
Introduction to Dynamic PET

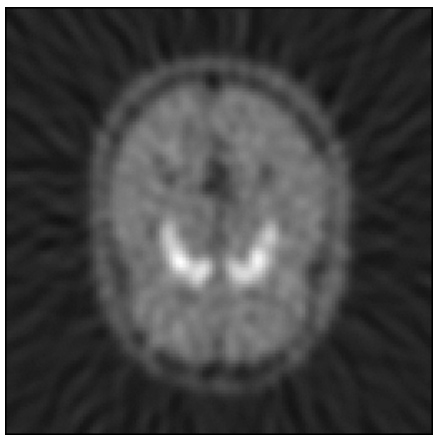


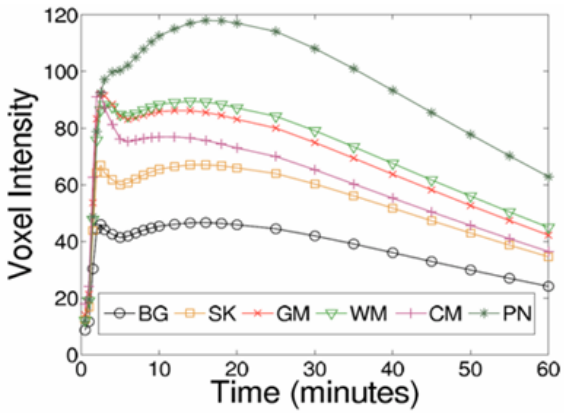


Introduction to Dynamic PET

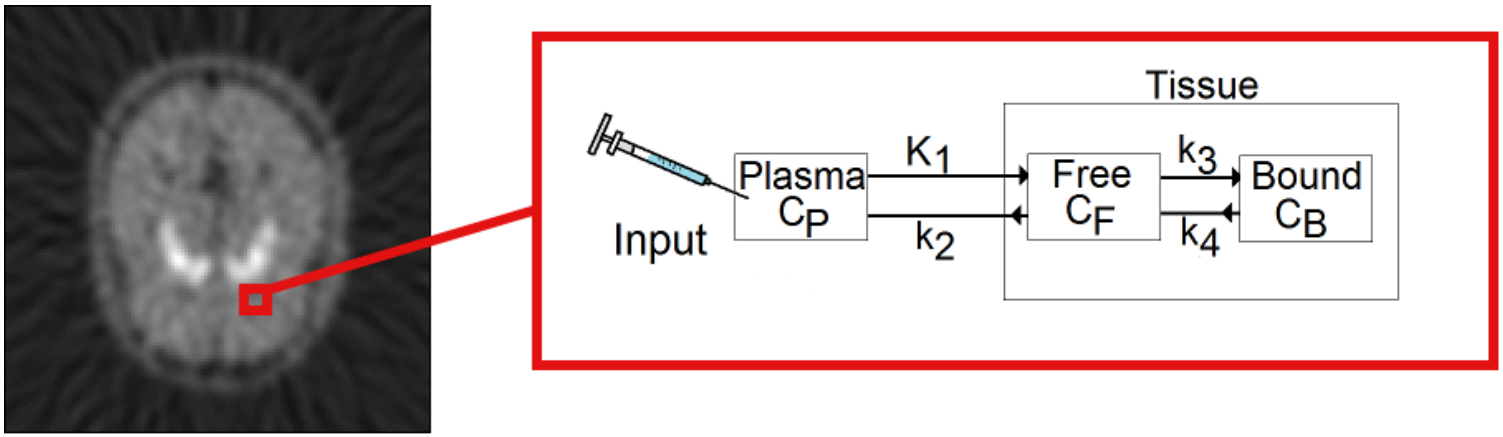




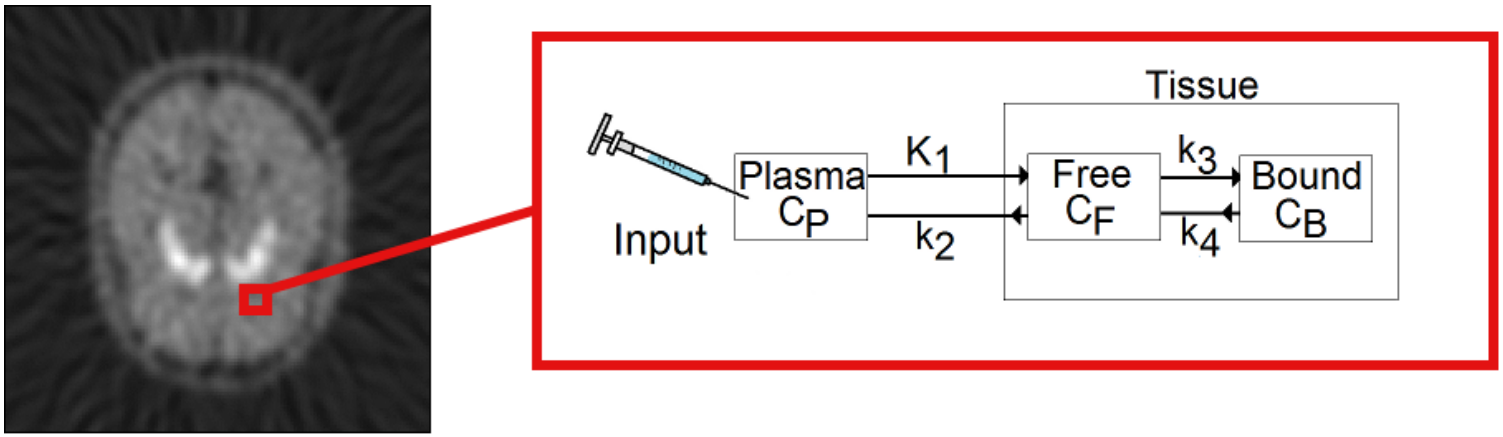




Kinetic Modelling

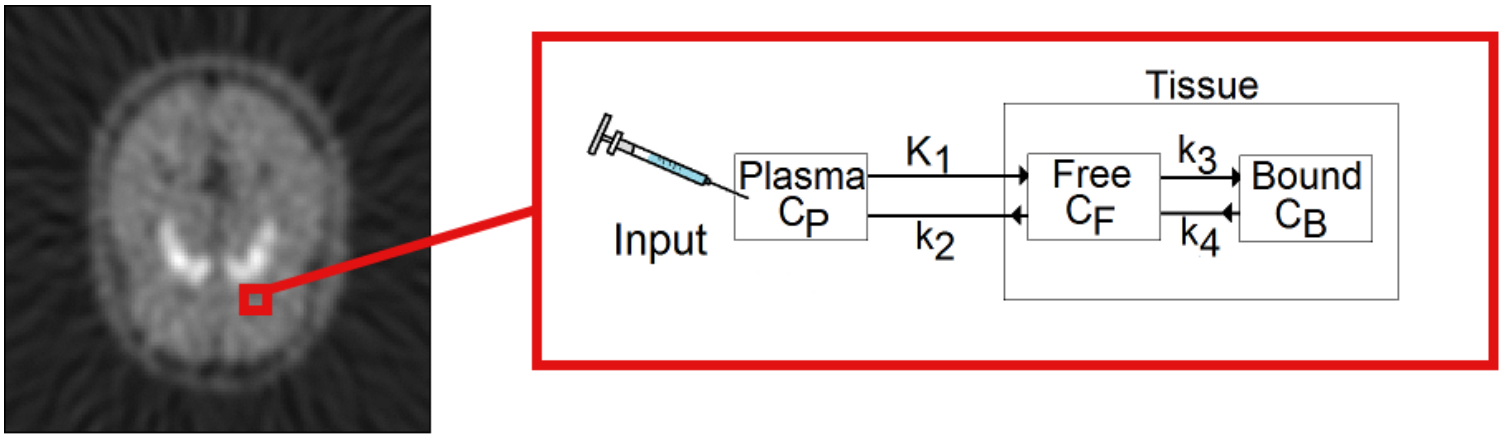


Kinetic Modelling



$$\begin{aligned} \frac{dC_F}{dt} &= K_1 C_P - k_2 C_F - k_3 C_F + k_4 C_B - \lambda C_F, \\ \frac{dC_B}{dt} &= k_3 C_F - k_4 C_B - \lambda C_F. \end{aligned} \tag{1}$$

Kinetic Modelling



$$\begin{aligned} \frac{dC_F}{dt} &= K_1 C_P - k_2 C_F - k_3 C_F + k_4 C_B - \lambda C_F, \\ \frac{dC_B}{dt} &= k_3 C_F - k_4 C_B - \lambda C_F. \end{aligned} \tag{1}$$

$$GMR = \frac{k_1 k_3}{k_2 + k_3} \propto \text{metabolism} \tag{2}$$

Fitting Kinetic Models

According to Gunn et al., 2001 [2], the solution to these ODEs is:

$$\underbrace{O(t)}_{\substack{\text{Output}\\ \text{function}}} = \underbrace{\sum_{i=1}^{C} C_i(t)}_{\substack{\text{Sum of}\\ \text{concentrations}}} = \left[\underbrace{I_f}_{\substack{\text{Input}\\ \text{function}}} * \underbrace{\left(\sum_{i=1}^{C} \varphi_i e^{-\theta_i t} \right)}_{\substack{\text{Response}\\ \text{function}}} \right](t). \tag{3}$$

Fitting Kinetic Models

According to Gunn et al., 2001 [2], the solution to these ODEs is:

$$\underbrace{O(t)}_{\substack{\text{Output}\\ \text{function}}} = \underbrace{\sum_{i=1}^{C} C_i(t)}_{\substack{\text{Sum of}\\ \text{concentrations}}} = \left[\underbrace{I_f}_{\substack{\text{Input}\\ \text{function}}} * \underbrace{\left(\sum_{i=1}^{C} \varphi_i e^{-\theta_i t} \right)}_{\substack{\text{Response}\\ \text{function}}} \right](t). \tag{3}$$

Note: φ_i, θ_i are explicitly related to k_i.

Fitting Kinetic Models

According to Gunn et al., 2001 [2], the solution to these ODEs is:

$$\underbrace{O(t)}_{\substack{\text{Output}\\ \text{function}}} = \underbrace{\sum_{i=1}^{C} C_i(t)}_{\substack{\text{Sum of}\\ \text{concentrations}}} = \left[\underbrace{I_f}_{\substack{\text{Input}\\ \text{function}}} * \underbrace{\left(\sum_{i=1}^{C} \varphi_i e^{-\theta_i t} \right)}_{\substack{\text{Response}\\ \text{function}}} \right](t). \tag{3}$$

Note: φ_i, θ_i are explicitly related to k_i.

Recover kinetic parameters at each voxel x by minimization:

$$\min_{\theta,\phi} \sum_{t=0}^{T} \left[O(t) - I(x,t) \right]^2 \tag{4}$$

Challenges of Fitting Kinetic Models

Implications of this least squares problem:

Challenges of Fitting Kinetic Models

Implications of this least squares problem:

- Kinetic models are nonlinear (exponential terms)

Challenges of Fitting Kinetic Models

Implications of this least squares problem:

- Kinetic models are nonlinear (exponential terms)
- Simple optimization is insufficient: typically, Levenberg-Marquardt algorithm is used

Challenges of Fitting Kinetic Models

Implications of this least squares problem:

- Kinetic models are nonlinear (exponential terms)
- Simple optimization is insufficient: typically, Levenberg-Marquardt algorithm is used
- Available software, COMKAT (Muzik et al., 2001 [1]) compartmental modelling tookit, takes seconds per fitting of a single TAC.

Observations

Why is COMAT slow?

- Performs numerical convolution of finely sampled input function

Observations

Why is COMAT slow?

- Performs numerical convolution of finely sampled input function

- Computes numerical approximation of gradient and jacobian of fitting function

Observations

Why is COMAT slow?

- Performs numerical convolution of finely sampled input function
- $\Rightarrow$ With an analytic input function, the convolution can be solved analytically!
- Computes numerical approximation of gradient and jacobian of fitting function

Observations

Why is COMAT slow?

- Performs numerical convolution of finely sampled input function
- $\Rightarrow$ With an analytic input function, the convolution can be solved analytically!
- Computes numerical approximation of gradient and jacobian of fitting function
- $\Rightarrow$ With an analytic input function, the derivatives can be computed exactly!

Analytic Kinetic Fitting

Analytic input function (D. Feng et al., 2003 [3]):

$$I_f = \sum_{i=1}^{4} a_i e^{-\lambda_i t} \tag{5}$$

Analytic Kinetic Fitting

Analytic input function (D. Feng et al., 2003 [3]):

$$I_f = \sum_{i=1}^{4} a_i e^{-\lambda_i t} \tag{5}$$

Direct substitution into the output function, and the convolution simplifies:

$$\begin{aligned} O(t) &= \int_{\tau=0}^{t} \left[\sum_{i=1}^{4} a_i e^{-\lambda_i \tau} \right] \left[\sum_{i=1}^{C} \varphi_i e^{-\theta_i (t-\tau)} \right] \\ &= \sum_{i=1}^{C} \sum_{j=1}^{4} \left(\frac{a_j \theta_i}{\phi_i - \lambda_j} \left(e^{-\lambda_j t} - e^{-\phi_i t} \right) \right). \end{aligned} \tag{6}$$

Analytic Kinetic Fitting

Analytic input function (D. Feng et al., 2003 [3]):

$$I_f = \sum_{i=1}^{4} a_i e^{-\lambda_i t} \tag{5}$$

Direct substitution into the output function, and the convolution simplifies:

$$\begin{aligned} O(t) &= \int_{\tau=0}^{t} \left[\sum_{i=1}^{4} a_i e^{-\lambda_i \tau} \right] \left[\sum_{i=1}^{C} \varphi_i e^{-\theta_i (t-\tau)} \right] \\ &= \sum_{i=1}^{C} \sum_{j=1}^{4} \left(\frac{a_j \theta_i}{\phi_i - \lambda_j} \left(e^{-\lambda_j t} - e^{-\phi_i t} \right) \right). \end{aligned} \tag{6}$$

The derivatives are defined exactly as well, for use in the Levenberg-Marquardt algorithm!

Parallel Levenberg-Marquardt

To further increase performance, the Levenberg-Marquardt algorithm can be run in parallel on the GPU.

Parallel Levenberg-Marquardt

To further increase performance, the Levenberg-Marquardt algorithm can be run in parallel on the GPU.

- Each voxel fitting is independent

Parallel Levenberg-Marquardt

To further increase performance, the Levenberg-Marquardt algorithm can be run in parallel on the GPU.

- Each voxel fitting is independent
- Levenberg-Marquardt is an iterative algorithm
- Can implement algorithm in CUDA, and store algorithm state for each thread

Parallel Levenberg-Marquardt

To further increase performance, the Levenberg-Marquardt algorithm can be run in parallel on the GPU.

- Each voxel fitting is independent
- Levenberg-Marquardt is an iterative algorithm
- Can implement algorithm in CUDA, and store algorithm state for each thread
- Threads are kept synchronized by performing algorithm steps until all fittings in a block complete.

Overview of Experiments

We are interested in the speed and accuracy of the proposed changes, as compared to COMKAT. In order to have reliable ground truth, synthetic experiments were performed:

Overview of Experiments

We are interested in the speed and accuracy of the proposed changes, as compared to COMKAT. In order to have reliable ground truth, synthetic experiments were performed:

- TAC were generated according to a 2 compartment, FDG model, with kinetic parameters randomly selected from a realistic range.

Overview of Experiments

We are interested in the speed and accuracy of the proposed changes, as compared to COMKAT. In order to have reliable ground truth, synthetic experiments were performed:

- TAC were generated according to a 2 compartment, FDG model, with kinetic parameters randomly selected from a realistic range.
- Normally distributed noise was added to each TAC

Overview of Experiments

We are interested in the speed and accuracy of the proposed changes, as compared to COMKAT. In order to have reliable ground truth, synthetic experiments were performed:

- TAC were generated according to a 2 compartment, FDG model, with kinetic parameters randomly selected from a realistic range.
- Normally distributed noise was added to each TAC
- Initializations were created by perturbing the correct parameters.

Overview of Experiments

We are interested in the speed and accuracy of the proposed changes, as compared to COMKAT. In order to have reliable ground truth, synthetic experiments were performed:

- TAC were generated according to a 2 compartment, FDG model, with kinetic parameters randomly selected from a realistic range.
- Normally distributed noise was added to each TAC
- Initializations were created by perturbing the correct parameters.
- The amount of noise added, and degree of perturbation of the initialization, were each independentally varied.

Overview of Experiments

We are interested in the speed and accuracy of the proposed changes, as compared to COMKAT. In order to have reliable ground truth, synthetic experiments were performed:

- TAC were generated according to a 2 compartment, FDG model, with kinetic parameters randomly selected from a realistic range.
- Normally distributed noise was added to each TAC
- Initializations were created by perturbing the correct parameters.
- The amount of noise added, and degree of perturbation of the initialization, were each independentally varied.
- Four levels of each parameter were tested, with the other parameter fixed at a reasonable value.

Overview of Experiments

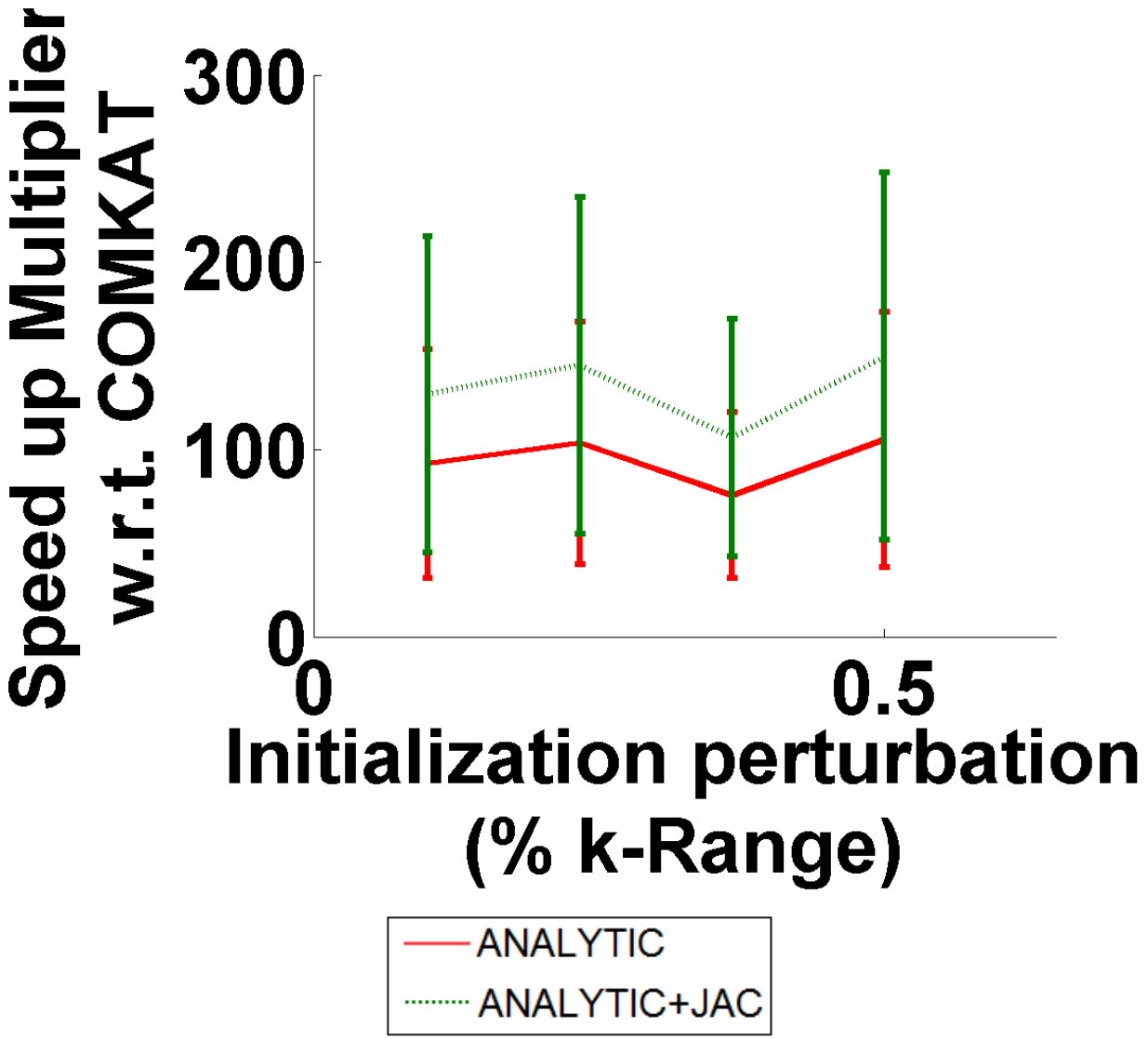
We are interested in the speed and accuracy of the proposed changes, as compared to COMKAT. In order to have reliable ground truth, synthetic experiments were performed:

- TAC were generated according to a 2 compartment, FDG model, with kinetic parameters randomly selected from a realistic range.
- Normally distributed noise was added to each TAC
- Initializations were created by perturbing the correct parameters.
- The amount of noise added, and degree of perturbation of the initialization, were each independentally varied.
- Four levels of each parameter were tested, with the other parameter fixed at a reasonable value.
- For each level tested, 20 repetitions were performed with different realizations of noise and perturbation.

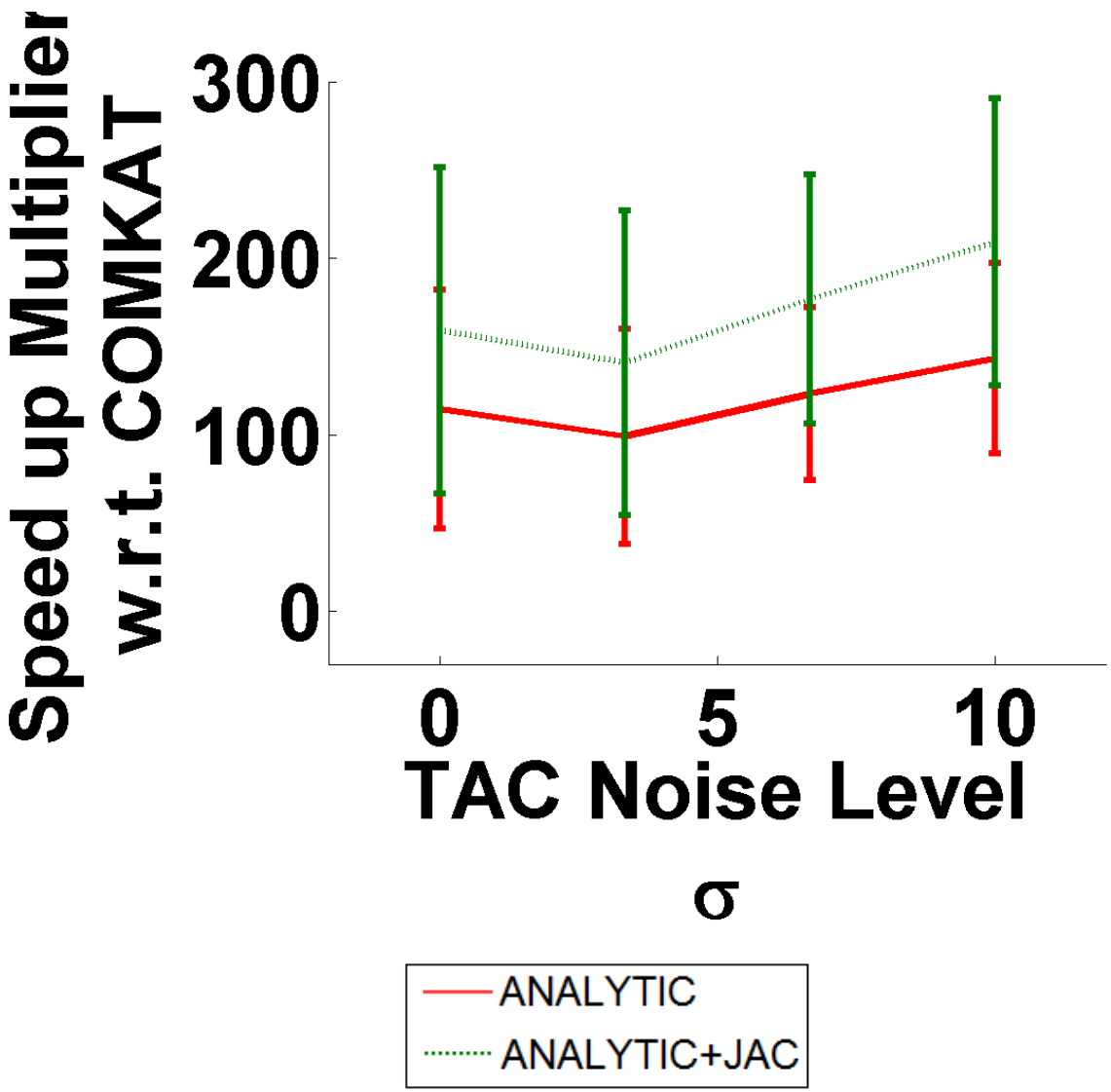
Compared Methods

- ANALYTIC: Analytic output function, approximated derivatives
- ANALYTIC+JAC: Analytic output function, analytic derivatives
- COMKAT: COMKAT fitting tool
- CUDA: Parallel GPU implementation of ANALYTIC+JAC

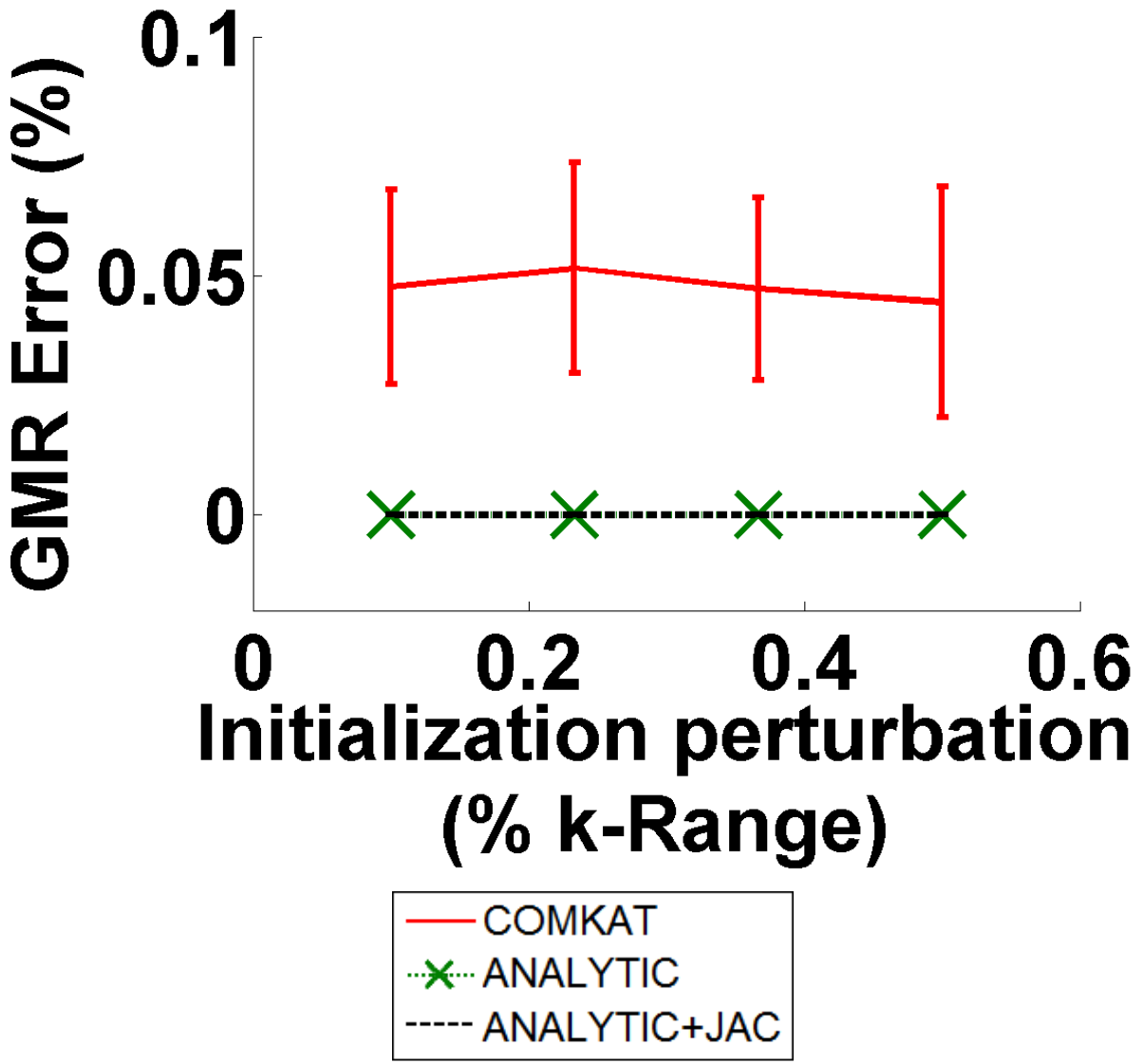
Speedup vs. Perturbation



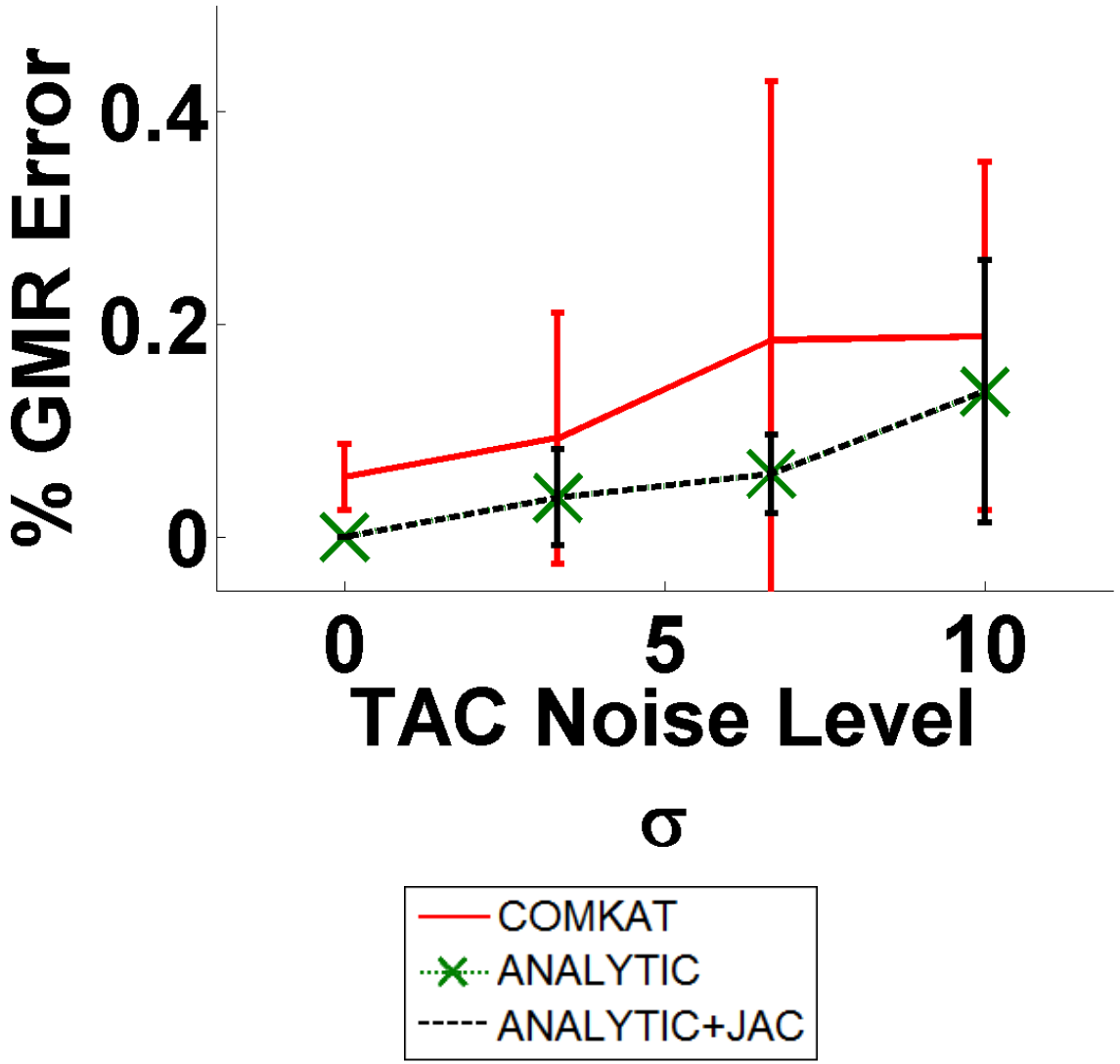
Speedup vs. Noise



Error vs. Perturbation



Error vs. Noise



Compare Parallelism

To test the effect of increased CUDA parallelism, five sets of 10,000 synthetic TACs were generated with different noise and initizliation realizations.

Compare Parallelism

To test the effect of increased CUDA parallelism, five sets of 10,000 synthetic TACs were generated with different noise and initizliation realizations.

- Level of parallelism was varied from 1 to 10,000

Compare Parallelism

To test the effect of increased CUDA parallelism, five sets of 10,000 synthetic TACs were generated with different noise and initizliation realizations.

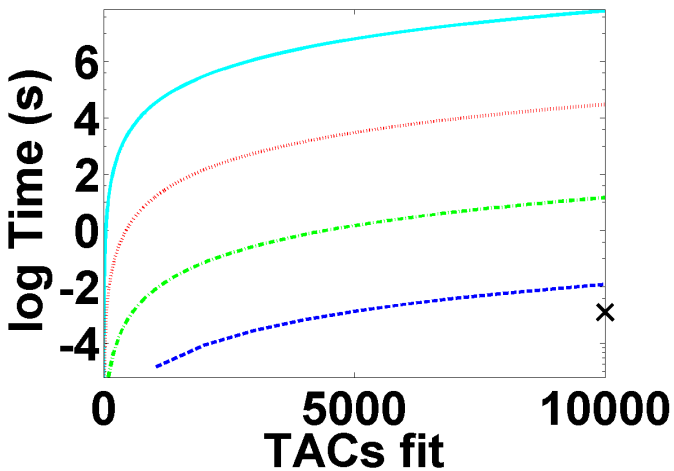
- Level of parallelism was varied from 1 to 10,000
- Execution time was recorded at each level

Compare Parallelism

To test the effect of increased CUDA parallelism, five sets of 10,000 synthetic TACs were generated with different noise and initizliation realizations.

- Level of parallelism was varied from 1 to 10,000
- Execution time was recorded at each level
- Prediction: For each factor 10 increase in parallelism, speed of fitting should increase by 10

Parallelism Results



	Parallelism	Speed (TAC/s)
———	1	45
·····	10	450
-·-·-	100	4,505.6
- - -	1,000	33,758.4
×	10,000	74,906.4

Conclusion

Conclusions

- Analytic methods demonstrate speed-up of approximately $100\times$ over COMKAT, with equivalent accuracy.
- Performing fitting in parallel with CUDA furthered increased performance to $130,000\times$.
- This allows a typical PET volume of $128 \times 128 \times 63$ to be fit in approximately 13.8 seconds, as compared to 21.5 days using COMKAT.

Conclusion

Conclusions

- Analytic methods demonstrate speed-up of approximately $100\times$ over COMKAT, with equivalent accuracy.
- Performing fitting in parallel with CUDA furthered increased performance to $130,000\times$.
- This allows a typical PET volume of $128 \times 128 \times 63$ to be fit in approximately 13.8 seconds, as compared to 21.5 days using COMKAT.

Future Work

- Investigate parallel spatial regularization of fitting
- Explore application to other modalities (DCE-MRI)

References

Thank you for your time! Questions?

Presented by: Ahmed Saad - aasaad@cs.sfu.ca

http://mial.fas.sfu.ca/

References

1. R. Muzic et al., COMKAT: compartment model kinetic analysis tool. *J. Nucl. Med.*, 42(4):636-645, 2001.
2. R.N. Gunn et al., PET compartmental models. *J. Cereb. Blood Flow and Metab.*, 21(6):635-652, 2001.
3. D. Feng et al., Models for computer simulation of input functions for tracer kinetic modeling with PET. *Int. Bio. Comp.*, 32:95-110, 1993.