GPGPU によるバイオロジカルパスウェイモデルの高速なパラメータ推定

予測発見戦略研究センター　データ同化グループ
林　圭佐　（JST/CREST）

1　はじめに

データ同化は気象・海洋シミュレーションの分野で発達してきた手法である。完全ではないモデルと観測されたデータを融合することで、現実に近づけた計算結果を得るほか、適切な初期値・パラメータの推定にも用いられる。その有用性は広く認知され、気象・海洋の分野のみならず宇宙空間、ゲノム情報と分野を超えた広がりが見られる(樋口(2007))。

ゲノム情報におけるデータ同化(Nagasaki et al.(2006))の応用例のひとつは、常微分方程式で記述される生化学反応ネットワークと反応濃度の時系列データを利用して生化学反応ネットワークのパラメータ推定を行うものである。このとき、求めるべきパラメータの次元は数十から数千のオーダーで高次元となる。問題の空間が高次元になると問題空間の体積は指数関数的に増加するため、パラメータ推定は困難を伴う。問題によっては、背景やモデルに応じた制限を加えることで問題空間を限定するアプローチが有効になるかもしれない。本報では、粒子フィルタの並列計算向くアルゴリズムに着目し、近年発達の著しい超並列コンピューティングに適合するようなアプローチにより、高次元問題への応用の最初の手かがりとしたい。

2　ベイズ推定によるパラメータ推定

次のような非線型状態空間モデルで記述した問題を考える。

システムモデル：　$x_t = f(x_{t-1}, v_t), v_t \sim p(v_t)$.

観測：　$y_t = h(x_t) + w_t, w_t \sim p(w_t)$.

初期値を含めたモデルパラメータ：$\tilde{\theta} \sim p(\tilde{\theta})$.

以上で定義される状態空間モデルにおいて、与えられた観測 $\{y_t\}_{t=1,2,\ldots,T}$ のもとで、初期値を含めたモデルパラメータ $\tilde{\theta}$ の事後分布　$p(\tilde{\theta}|y_{1:T})$ を推定する。

ベイズの定理から、$\tilde{\theta}$ の事後分布は事前分布と尤度の積に比例する。

$$p(\tilde{\theta}|y_{1:T}) \propto p(\tilde{\theta}) p(y_{1:T}|\tilde{\theta}) \tag{1}$$

式(1)の右辺は尤度の分解の公式を用いて

$$p(\tilde{\theta}) p(y_{1:T}|\tilde{\theta}) = p(\tilde{\theta}) \prod_{j=1}^{T} p(y_j | y_{1:j-1}, \tilde{\theta}) \tag{2}$$

と書くことができる。

式(2)の右辺のうち、事前確率分布 $p(\tilde{\theta})$ は研究者の先験情報から与える。一時点尤度は、観測誤差の確率分布と観測時刻　t-1　までの観測データを同化した予測確率分布から求まる。

$$p(y_t | y_{1:t-1}, \tilde{\theta}) = \int p(y_t | x_t, \tilde{\theta}) p(x_t | y_{1:t-1}, \tilde{\theta}) dx_t .$$

3　Sequential Importance Sampling

既報(Nakamura et al. (2009)) では、観測データを同化した確率分布を求めるために、粒子フィルタ(SIR)を用いた。粒子フィルタでは、実現値の集合でモンテカルロ近似された状態変数の分布に対し、一時点尤度に基づくリサンプリングを実施して観測データの情報を取り込む。リサンプリングには、実現値集合と観測から求めた尤度の合計を確定する必要があり、場合により通信が発生する。尤度の合計と粒子情報交換に伴い必要とする通信が並列計算機の性能低下を引

き起こす。本報では Sequential Importance Sampling (SIS)を用いて対処する。SIS型粒子フィルタではリサンプリングの代わりに、尤度に基づいた重みを用いる。SISで同化すると、予測分布は

$$p\left(x_t \middle| y_{1:t-1}, \tilde{\theta}\right) \cong \sum_{i=1}^{N} \omega_{t-1|t-1}^{(i)} \delta\left(x_t - x_{t-1}^{(i)} | t\right), \tag{3}$$

$$\omega_{t-1|t-1}^{(i)} = \frac{\omega_{t-2|t-2}^{(i)} p\left(y_{t-1} \middle| x_{t-1|t-2}^{(i)}, \tilde{\theta}\right)}{\sum_{j=1}^{N} \omega_{t-2|t-2}^{(j)} p\left(y_{t-1} \middle| x_{t-1|t-2}^{(j)}, \tilde{\theta}\right)}, \tag{4}$$

で近似される。ここで δ はDirac のデルタ関数、N はモンテカルロ近似に用いる粒子数である。重みの分母の計算は、全粒子にわたる縮約が必要であるためオーバーヘッドが大きい。並列処理中の式(4)の分母を確定せずに非同期に実装する。各観測時刻で尤度を保存し、全粒子の計算後にまとめて尤度関数の値を計算するようにすると、式(2)は

$$p\left(\tilde{\theta}\right) \prod_{j=1}^{T} p\left(y_j \middle| y_{1:j-1}, \tilde{\theta}\right) = p\left(\tilde{\theta}\right) \frac{1}{N} \sum_{i=1}^{N} \prod_{t=1}^{T} p\left(y_t \middle| x_{1:t}^{(i)}, y_{1:t-1}, \tilde{\theta}\right), \tag{4}$$

となることが示せる。

式(4) から、事後確率分布は、観測データの条件付き確率の加重平均で表される。つまり、モンテカルロ近似に用いる粒子に関して、それぞれ独立に計算することが可能であるし、結果として粒子を逐次的に増加可能である。 粒子の増加に用いるデータは、尤度 $s^N = \frac{1}{N} \sum_{i=1}^{N} \prod_{t=1}^{T} p\left(y_t \middle| x_{1:t}^{(i)}, y_{1:t-1}, \tilde{\theta}\right)$ と粒子数Nだけで良いので、省メモリであり、通信量を低く抑えることができる。N粒子とM粒子による結果を合算するには次式を用いる。

$$L^{N+M}\left(\tilde{\theta}\right) = \frac{N}{N+M} L^{N}\left(\tilde{\theta}\right) + \frac{M}{N+M} L^{M}\left(\tilde{\theta}\right).$$

粒子追加の処理により、適切な精度で収束するまで粒子数を増加可能なので、このように粒子を増大して収束計算するSISの構成を「無限SIS」と呼ぶことにする。本手法は各粒子の計算が独立であり、通信量が少ないためスケーラビリティの高い手法であることが期待できる。

4　数値実験

文献(Nakamura et al. (2009))で用いられたサーカディアンリズムモデルのパラメータ推定問題を例題に数値実験を行う。CPU に Opteron Processor 2220 を搭載したPC に Tesla C870 を接続し、本手法を適用した。結果、10億粒子を使うった解析の計算時間は7時間45分だった。

5　まとめ

本手法において各粒子の計算は独立であり、状態変数を高速なメモリに置いて計算できるため、GPUで計算する例題としては理想的だったといえる。「無限SIS」は各粒子が同期不要で独立に計算でき、シミュレーション中にはいかなる通信も発生しない。そのため GPU で能率的に計算することができる。これまで数日かかっていたような計算が1日で済むようになった。

謝辞　この研究は明治大学 中村和幸特任講師，統計数理研究所 吉田亮助教，東京大学 医科学研究所 長崎正朗助手，宮野悟教授らから基本となるソフトウェアの提供を受けました。厚く御礼申しあげます。

参考文献

樋口知之(2007).「全体モデルから局所モデルへ/状態空間モデルとシミュレーション」、数学セミナーII、Vol. 46, No. 11, 30-36.

Nagasaki, M. et al. (2006). Genomic Data Assimilation for Estimating Hybrid Functional Petri Net from Time-course Gene Expression Data, Genome Informatics, Vol. 17, 46-61.

Nakamura, K. et al. (2009). Parameter estimation of in silico biological pathways with particle filtering towards a petascale computing, Pacific Symposium on Biocomputing. 14: 227-238.