Vol.2013-ARC-207 No.18
Vol.2013-HPC-142 No.18
2013/12/16

統計的機械学習を用いたSIMD演算性能の推定

緒方 健太[1]　Trouvé Antoine[2]　眞木 淳[2]　Clarke Hadrien[2]　村上 和彰[3]

概要：近年，HPC向け高速化機構としてSIMDの活用が一般的になっているが，その効果は対象とするHPCプログラムの特性，データサイズ，データ配置，使用するコンパイラの能力，プログラムを実行するプラットフォームの命令セットアーキテクチャおよびマイクロアーキテクチャ等に依存して大きく異なる．本稿では，統計的機械学習に基づくSIMD性能推定手法を検討する．入力データサイズ一定のテンソル積計算プログラムを対象とした場合には線形回帰に基づく性能モデルが適用可能であることを実験により明らかにした．

キーワード：HPC，SIMD，機械学習，性能推定

SIMD Performance Estimation using Statistical Machine Learning

Kenta Ogata[1]　Antoine Trouvé[2]　Jun Maki[2]　Hadrien Clarke[2]　Kazuaki Murakami[3]

Abstract: In recent years, the use of SIMD has become common as faster mechanism for HPC. However, the degree of the effect is different because of charcteristics of the target HPC program, data size, data allocation, instruction set architecture and microarchitecture in the platform. In this paper, we examine SIMD performance estimation using machine lerning, and we revealed that linear regression applies to performance model of tensor calculus program.

Keywords: HPC, SIMD, Machine Learning, Performance Estimation

1. はじめに

近年，HPCの高速化機構として，SIMD（Single Instruction Multiple Data）が一般的に利用されている．SIMDとはアプリケーションのデータレベル並列性を活用し，高速化を図る技術である．このSIMD演算性能はプログラムの入力行列の次元サイズや配列の添え字変数のパターンによって大きく異なるため，よりよい性能を達成するためにも，性能推定が重要になる．さらに，近年のコンピュータシステムの複雑化により，従来の手法のようにコンピュータシステムの動作の詳細を明示的に性能モデルに組み込んだり，シミュレータを構築したりすることで性能推定を行う事は困難である．

そこで，統計的機械学習を利用し，コンピュータシステムの動作の詳細を明示的に性能モデルに組み込むことなく，SIMD演算性能を推定する手法を検討する．この手法では，入力行列の次元サイズや配列の添え字変数の取り方といったワークロードの特徴を機械学習の入力とし，さらに有効な学習アルゴリズムを設定することで，SIMD演算の性能モデルを生成する．生成した性能モデルを用いて，SIMD演算性能を推定する．

統計的機械学習を用いたSIMD演算性能の推定手法において，どんな機械学習アルゴリズムが効果的であるか，機械学習にはどのような入力が必要かということを明らかにするために，コンピュータシステム上のワークロードの特徴を静的特徴（SWC：Static Workload Characteristics）と動的特徴（DWC：Dynamic Workload Characteristics）の2つに分類し，性能推定に有効な特徴を調査する．また，

[1] 九州大学大学院 システム情報科学府
Kyushu University, Japan
[2] （公財）九州先端科学技術研究所
ISIT, Momochihama, Fukuoka 814–0001, Japan
[3] 国立大学法人九州大学 大学院システム情報科学研究院
情報知能工学部門，Kyushu University, Japan

Vol.2013-ARC-207 No.18
Vol.2013-HPC-142 No.18
2013/12/16

SWC と DWC のそれぞれを入力として，性能推定またはある性能グループに分類する性能モデルを生成するために有効な学習アルゴリズムを調査する．

本研究は，オブジェクトコード（O）から SWC の抽出及び SWC を用いたバッチ学習による性能モデリングを行う（SWC-O）．アセンブリコードから静的命令カウントを抽出した後で，統計的機械学習によりプログラムの実行時間を推定するモデルを生成する．具体的には，SIMD 演算を利用したプログラムの実行時間の性能モデルを，学習アルゴリズムに線形回帰を用いることで生成する．本稿では，テンソル積計算の配列の添え字を変更したプログラムを対象として，得られた性能モデルによる推定値と実際の計算結果の性能値との比較を行い，線形回帰の学習アルゴリズムが妥当かどうか評価する実験を行う．その結果，線形回帰の線形回帰式の自由度調整済み決定係数は 0.94 と，高い精度が得られた．また t 検定や F 検定からも，生成したモデルは有意であるとの結果となった．

本稿の構成は以下の通りである．第 2 章では SIMD に関わる演算と従来手法の問題について述べる．第 3 章では統計的機械学習をベースとした性能モデリングについて述べる．第 4 章では性能モデリング及び生成したモデルの妥当性検証実験について述べ，第 5 章でまとめる．

2. SIMD に関わる演算の推定と従来手法の問題

SIMD（Single Instruction Multiple Data）とは，1 つの命令で複数のデータを並列に処理することで高速化を図る技術であり，今日の HPC（High Performance Computing）分野では高速化技術として SIMD の活用が一般的となっている．しかし，この SIMD 演算の実効性能にはばらつきがある．

例として，図 1 のようなテンソル積計算プログラムを考える．この配列 A，B，C の添え字である「*」は i，j，k，l のどれかを取る．図 2 は，配列 A，B，C が添え字変数に i，j，k，l のどれを取るか，そして入力行列次元サイズ（データサイズ）N の値をどう設定するかで，SIMD 演算の性能（Operations / clock-cycle）が大きく異なることを示している．ここで Operations とは最内ループのボディのステートメントの実行回数であり，4 重ループなので全体的に N^4 となる．この図において，ij-jlk-kil 等の表記はテンソル積プログラムにおける配列 A，B，C の添え字変数を表している．例えば，ij-jlk-kil は A[i][j]，B[j][l][k]，C[k][i][l] を表す．このように，SIMD 演算を利用し，より良い性能を達成しようとするとき，配列の添え字変数，入力行列の次元サイズを変化させたとき性能はどう変化するか，SIMD 演算によるスピードアップはどの程度であるのかを推定することは重要である．

従来の性能推定手法にはコンピュータシステムの動作を

```
for(i = 0; i < N; i++)
  for(j = 0; j < N; j++)
    for(k = 0; k < N; k++)
      for(l = 0; l < N; l++)
        A[*][*] += B[*][*][*] × C[*][*][*]
```

図 1 テンソル積計算プログラム

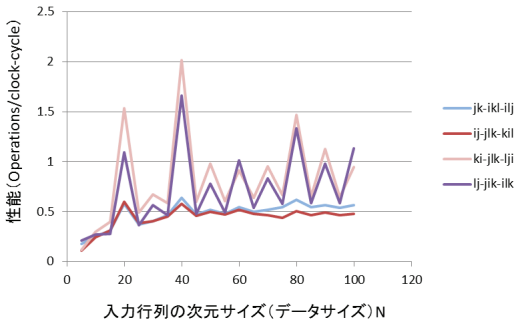

図 2 テンソル積プログラムにおける SIMD 演算性能の変化

理解し，人手で性能モデル式を生成するといった手法 [1] や，シミュレータを構築して性能推定するといった手法 [7] がある．しかし，今日のコンピュータシステムは複雑化しており，そのためシステムの動作に基づき人手で性能モデルを生成したりシミュレータを構築することは困難である．

3. 統計的機械学習に基づく性能推定

2 章で取り上げた課題を解決するために，人手で性能モデルを構築することのない，統計的機械学習に基づく性能推定を行う．3.1 節では，性能推定に機械学習を利用する際の課題，また機械学習に基づく性能推定の目標を述べ，3.2 節では，この性能推定の概要を述べる．

3.1 統計的機械学習に基づく性能推定の目標

機械学習とは，ある現象の特徴データをコンピュータに入力することで，その入力データをもとに学習アルゴリズムに従って，その特徴に関わるパラメータを推論させ，対象のモデルを自動的に生成する技術である．機械学習を用いることにより，複雑なコンピュータシステムの構成および動作に基づいた性能モデルを人手で構築することなく性能推定を行うことができる．しかし，学習に用いる入力データや学習アルゴリズムについて何を選択するかといった課題がある．

そのために，学習の入力データに用いるワークロード特徴の中から性能推定に有効な特徴や，性能モデリングに有効な学習アルゴリズムを明らかにすることを目標とする．

コンピュータシステム上のワークロードの特徴は，ワークロード実行前に抽出可能な静的特徴（SWC：Static Workload Characteristics）と，実行することで抽出可能となる動的特徴（DWC：Dynamic Workload Characteristics）の 2 つに分類される．さらに，SWC と DWC のそれぞれに

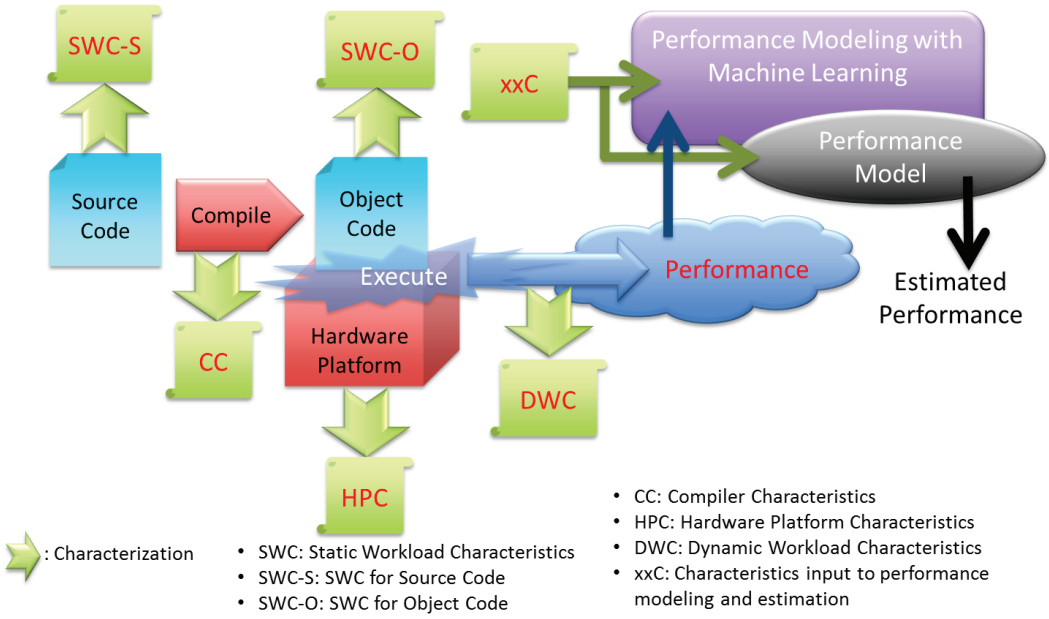

図 3 統計的機械学習に基づく性能推定のフロー

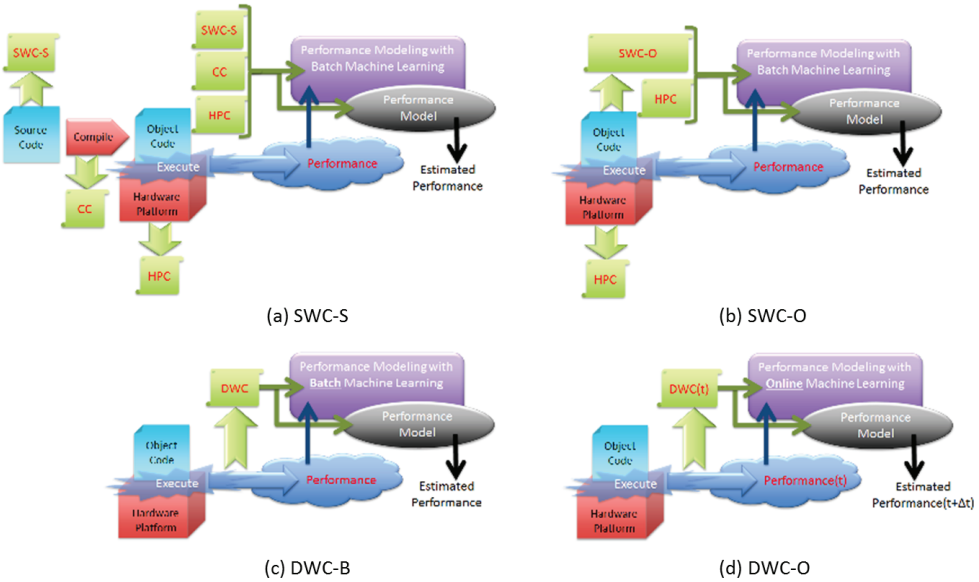

図 4 SWC-S，SWC-O，DWC-B，DWC-O それぞれに対する特徴抽出のフロー

ついて様々な特徴が定義できる．これら各種特徴から，真に有効な特徴を明らかにし，これらを自動抽出する技術を開発する．これにより抽出した特徴量を学習に用いる．

また，特徴量を入力とし，「性能の推定」や「ある性能グループに分類」を出力とするような性能モデルを統計的機械学習により生成する．機械学習にはバッチ学習及びオンライン学習の 2 種類の手法が存在し，それぞれに学習アルゴリズムが多く存在するため，これら性能モデルを用いた実験により，推定や分類に有効な学習アルゴリズムを明らかにする．

3.2 統計的機械学習に基づく性能推定の概要

図 3 は統計的機械学習に基づく性能推定のフローを示したものである．これは下記の 4 つに分類される．

- SWC-S：ソースコード（S）に対する静的特徴（SWC）の抽出や当該静的特徴を用いたバッチ学習による性能モデリング（図 4（a））
  - 入力行列の次元サイズや配列の添え字変数といった静的特徴から性能モデルを生成する．生成する性能モデルは，特定のコンパイラやハードウェアプラットフォームに依存しない「一般性能モデル」またはコンパイラやハードウェアプラットフォームに依存する「特殊性能モデル」の 2 種類である．本稿では

対象にしていない．先に特殊性能モデルの生成に取り組み，その後，一般性能モデルの生成に取り組む．

- SWC-O：オブジェクトコード（O）に対する静的特徴（SWC）抽出や当該静的特徴を用いたバッチ学習による性能モデリング（図 4（b））
  - アセンブリコードから抽出した命令カウント数といったコンパイラに依存する静的特徴から性能モデルを生成する．生成する性能モデルは，特定のハードウェアプラットフォームに依存しない「一般性能モデル」またはハードウェアプラットフォームに依存した「特殊性能モデル」の 2 種類である．先に特殊性能モデルの生成に取り組み，その後，一般性能モデルの生成に取り組む．
- DWC-B：動的特徴（DWC）抽出および当該動的特徴を用いたバッチ（B）学習による性能モデリング（図 4（c））
  - PCD（Performance Counter Data）のような動的特徴から回帰分析のようなバッチ学習により性能モデルを生成する．生成する性能モデルは，コンパイラやハードウェアプラットフォームに依存した「特殊性能モデル」である．本稿では対象にしていない．
- DWC-O：動的特徴（DWC）抽出および当該動的特徴を用いたオンライン（O）学習による性能モデリング（図 4（d））
  - 動的特徴および性能を，$DWC(t)$ や $Performance(t)$ のようにある時間 t の関数として捉え，時間 Δt 後の性能 $Performance(t+\Delta t)$ を推定する．生成する性能モデルは，コンパイラやハードウェアプラットフォームに依存した「特殊性能モデル」である．本稿では対象にしていない．

本研究は，上記の 4 つの内の「SWC-O」にあたる性能モデリングを行う．生成した性能モデルにより SIMD 演算性能を推定する．

4. SWC-O における性能モデリング

本章では，SWC-O に対する性能モデリングにもとづき，モデル生成に用いる学習アルゴリズムや，実際に生成された性能モデルを説明する．

4.1 線形重回帰分析

本研究では，モデルを生成する際に線形重回帰分析を用いる．回帰分析とは，説明変数を用いて，目的変数を説明するモデルをデータから求めるデータ分析手法であり，直線関係でモデル化する方法を線形回帰分析と呼ぶ．目的変数とは，推定する変数のことを指し，説明変数は目的変数を説明する変数のことを指す．線形回帰分析において説明変数が 2 つ以上ある場合には，線形重回帰分析と呼ばれる．目的変数 y，説明変数 x_i，定数 a_i を用いて，線形重回帰分析では式 1 のような関係を測定したデータから推定する．

$$y = a_0 + \sum_{i=1}^{n} a_i * x_i \tag{1}$$

定数 a_i は最小二乗法により求める．最小二乗法とは推定値と実測値との差の平方和を最小にするという方法である．

4.2 SIMD 演算性能のモデリング

本研究では，オブジェクトコードから得られる静的特徴として静的命令カウントを用いる．静的命令カウントとはアセンブリファイルから抽出した命令のそれぞれの数のことを指す．静的命令カウントを用いて SIMD 演算を利用したプログラムの実行時間を線形重回帰分析によりモデル化する．このとき，線形重回帰分析の目的変数には実行時間，説明変数には静的命令カウントを設定する．

対象とするプログラムを図 5 に示す．機械学習の入力データとして，使用した行列の次元サイズ N を一定（$N = 100$）として，配列の添え字変数の取り方をそれぞれ変更した際の 443 組の実行時間及び静的命令カウントを利用した．オブジェクトコードの生成の際には ICC version12.1.5 の最適化オプション O2 及び xSSE3 を用いた．

実行時間は PAPI（Performance Application Programming Interface）[6] により計測する．配列の添え字変数が同じパターンにおいて 5 回計測し，この平均値を実行時間とする．表 2 は実行時間の計測を行った環境をまとめたものである．

線形重回帰分析は統計ソフトである R version 3.0.1[4] を用いる．説明変数の選択は R で定義されている step 関数を用いて，変数減少法により行う．変数減少法とは AIC（赤池情報量規準）[3] と呼ばれる指標により説明変数を決定する方法である．AIC が小さい値であるほど良いモデルとされるので，変数減少法では，AIC を一番減少させるように説明変数を 1 つずつ取り除いていく．この方法により，movss，unpcklps，mulps，addps，addq，xorl の静的命令カウントを説明変数とするように決定した．

実際に作成した実行時間のモデルを式 2 に示す．ここで，実行時間を T，movss，unpcklps，mulps，addps，addq，xorl の静的命令カウントをそれぞれ C_{movss}，$C_{unpcklps}$，C_{mulps}，C_{addps}，C_{addq}，C_{xorl} と表す．また，説明変数の係数を a で表しており，係数の値は表 2 に示す．

$$\begin{aligned} T = a_0 \ & + a_1 * C_{movss} + a_2 * C_{unpcklps} \\ & + a_3 * C_{mulps} + a_4 * C_{addps} \\ & + a_5 * C_{addq} + a_6 * C_{xorl} \end{aligned} \tag{2}$$

次章では，生成した SIMD 演算性能モデルについての妥当性検証を行う．

Vol.2013-ARC-207 No.18
Vol.2013-HPC-142 No.18
2013/12/16

表 1　プログラムの実行時間の測定環境

CPU	Core2 Extreme
コア数	4
L1 キャッシュ	32KB，8way
L2 キャッシュ	4MB，16way
キャッシュラインサイズ	64B

```
N = 100;
float A[N][N];
float B[N][N][N];
float C[N][N][N];

for(i = 0; i < N; i++)
  for(j = 0; j < N; j++)
    for(k = 0; k < N; k++)
      for(l = 0; l < N; l++)
        A[*][*] += B[*][*][*]×C[*][*][*]
```

図 5　性能モデル生成に用いるプログラム

表 2　係数の値

a_0	48657.2
a_1	27355.8
a_2	−24965.7
a_3	14698.0
a_4	−33023.6
a_5	2735.3
a_6	−3470.7

5. 性能モデルの妥当性検証実験

5.1 実験内容

4 章の方法により得られた実行時間のモデルから得た推定値が実際の実行時間をどの程度再現可能であるかを評価する．再現の検証の際にはモデル生成に使用したのと同一のプログラムを使用し，添え字入れ替えについても同じ変更を加える．評価結果から，学習アルゴリズムとして線形重回帰分析は有効であるか，静的特徴として静的命令カウントは有効であるかを考察する．

生成した実行時間のモデルによる予測値が実測値にどれほど当てはまっているかを示す指標には，自由度調整済み決定係数を用いる [2]．決定係数 R^2 は式 3 で定義され，実測値の変動をモデル式による推定値がどれほどの割合で説明できているかを表すものである．$\hat{y}$ は目的変数 y の得られたモデルから計算された推定値であり，$\bar{y}$ は目的変数の実測値の平均値である．決定係数は 1 に近い値になるほど，モデルが実測値をよく説明していると考えられる．

$$R^2 = 1 - \frac{\sum_{j=1}^{n}(y_j - \hat{y_j})^2}{\sum_{j=1}^{n}(y_j - \bar{y})^2} \quad (3)$$

決定係数は，モデルの説明変数の数が増えれば増えるほど，その値は大きくなり，再現性の良いモデルであると判断してしまう．そこで本稿では，説明変数の数の影響を除くために調整された決定係数（自由度調整済み決定係数）を用いた．標本数を N，自由度を p としたとき，自由度調整済み決定係数 $adjustedR^2$ は式 4 のように定義される．

$$adjustedR^2 = 1 - \frac{\sum_{j=1}^{n}(y_j - \hat{y_j})^2/(N-p-1)}{\sum_{j=1}^{n}(y_j - \bar{y})^2/(N-1)} \quad (4)$$

また，実行時間のモデル式について，それぞれの説明変数の係数の値が有意であるかどうかを判断するために，t 検定及び F 検定を行う [5]．

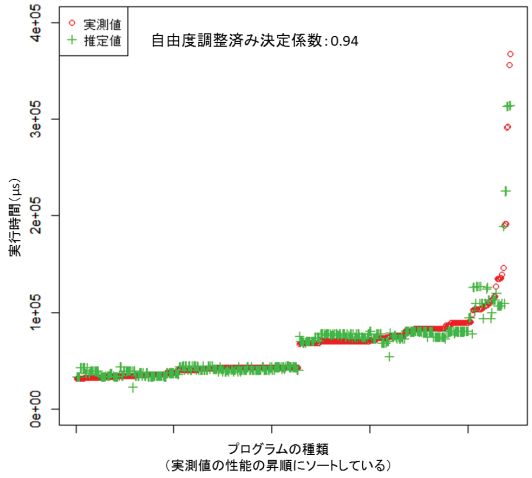

図 6　実行時間の予測値（緑）と実測値（赤）

5.2 実験結果

図 6 は実行時間のモデルによる予測値（緑）と実測値（赤）をプロットしたものである．図より，推定値が実測値に近い値となっていることがわかる．

実行時間のモデルは自由度調整済み決定係数が 0.94 と高い値を出しており，実測値の再現性が良いモデルだと考えられる．

表 3 は式 2 におけるそれぞれの係数の t 検定の結果及び F 検定の結果を示している．t 検定の結果より，全ての説明変数の係数において P 値は極めて小さいものとなっている．ゆえに，それぞれの係数が 0 となるという帰無仮説が棄却される．また，F 検定による P 値も小さいことがわかる．F 検定は，すべての係数が 0 であるという帰無仮説について検定を行っており，実験結果より帰無仮説は棄却される．t 検定，F 検定の結果から，このモデル式が有意であると考えられる．

以上の結果から，テンソル積計算の添え字入れ替え計算を対象とした場合，静的命令カウントから SIMD 演算の実行時間のモデルを生成する学習モデルとして，線形重回帰分析に基づく学習モデルを適用することが可能であるといえる．

6. 関連研究

機械学習を用いた性能推定の関連研究として，Stock らの研究 [8] が挙げられる．この研究では静的特徴から，ベクトル化した際の絶対性能を推定しているという点で，本

表 3 t 検定及び F 検定の結果

t 検定の結果		F 検定の結果
係数	P 値	P 値
a_0	$4.43 * 10^{-7}$	$2.20 * 10^{-16}$
a_1	$2.00 * 10^{-16}$	
a_2	$2.00 * 10^{-16}$	
a_3	$3.34 * 10^{-7}$	
a_4	$2.00 * 10^{-16}$	
a_5	$1.29 * 10^{-8}$	
a_6	$8.74 * 10^{-11}$	

研究と類似している．プログラムによっては性能推定の精度が良く，本研究のように静的特徴から性能の傾向を推定できると考えられる．しかし，本研究と異なり静的特徴はバイナリレベルの命令ミックスしか使用しておらず，コードレベルの特徴は使用していない．特にデータサイズを変化した際の性能の差を考慮していない．また最適化空間探索しかしておらず，回帰分析結果の誤差についても言及していない．本研究は研究 [8] と異なり，ソースコードレベルまたはオブジェクトコードレベルで性能推定を行い，またプログラムにおける入力データサイズの変化も考慮する予定である．

7. おわりに

入力行列のデータサイズや配列の添え字変数によって SIMD 演算性能は変化する．そこで，統計的機械学習を利用し，人手で性能モデルを構築することのなく性能推定を行う．本研究ではオブジェクトコードの特徴と線形回帰といったバッチ学習アルゴリズムから SIMD 演算における実行時間の性能モデリングを行った．テンソル積計算プログラムの添え字入れ替え計算を対象として，生成した性能モデルは，自由度調整済み決定係数は 0.94，t 検定においても有意であるという結果となった．この実験結果から，今回用いた学習アルゴリズムである線形回帰はテンソル積計算プログラムの添え字入れ替え計算の性能推定には有効であると考える．

今後の予定としては，今回の性能モデルの生成は入力行列の次元サイズは固定という条件で行っていたので，入力行列の次元サイズの変化による性能変化にも線形回帰の学習アルゴリズムが対応できるかを調査するつもりである．また，線形重回帰分析以外にも SIMD 演算性能推定に有効な学習アルゴリズムはないか調査するつもりである．

参考文献

[1] Baghsorkhi, S. S., Delahaye, M., Patel, S. J., Gropp, W. D. and Hwu, W.-m. W.: An adaptive performance modeling tool for GPU architectures, *SIGPLAN Notices*, pp. 105–114 (2010).
[2] Barrett, J. P.: The Coefficient of Determination - Some Limitation, *The American Statistician*, Vol. 28, pp. 19–20 (1974).
[3] Bozdogan, H.: Model selection and Akaike's Information Criterion (AIC): The general theory and its analytical extensions, *Psychometrika*, Vol. 52, pp. 345–370 (1987).
[4] GNU project: The Comprehensive R Archive Network, `http://cran.r-project.org/`.
[5] Harrell, Jr., F. E.: *Regression Modeling Strategies*, Springer New York (2001).
[6] Inovated Computing Laboratory: Performance Application Programming Interface, `http://icl.cs.utk.edu/papi/`.
[7] Meng, J., W. Sheaffer, J. and Skadron, K.: Robust SIMD: Dynamically Adapted SIMD Width and Multi-Threading Depth, *Parallel & Distributed Processing Symposium (IPDPS), 2012 IEEE 26th International*, pp. 107–118 (2012).
[8] Stock, K., Pouchet, L.-N. and Sadayappan, P.: Using Machine Learning to Improve Automatic Vectorization, *ACM Trans. Archit. Code Optim.*, Vol. 8, No. 4, pp. 50:1–50:23 (2012).