METODE *STEEPEST DESCENT* DENGAN UKURAN LANGKAH BARU UNTUK PENGOPTIMUMAN NIRKENDALA

D. WUNGGULI[1], B. P. SILALAHI[2], S. GURITMAN[3]

Abstrak

Metode *steepest descent* adalah metode gradien sederhana untuk pengoptimuman. Metode ini memiliki kekonvergenan yang lambat dalam menuju ke solusi optimum, hal ini terjadi karena langkahnya yang berbentuk zig-zag. Barzilai dan Borwein berusaha menyempurnakan metode ini dengan memodifikasi algoritme sehingga hasilnya berjalan cukup baik untuk masalah dengan dimensi yang besar. Hasil metode Barzilai dan Borwein ini telah memicu banyak penelitian pada metode *steepest descent*, diantaranya terdapat metode Alternatif Minimisasi dan metode Yuan. Dalam tulisan ini telah dimodifikasi metode *steepest descent* dengan ukuran langkah baru. Hasil modifikasi ini kemudian dibandingkan dengan metode Barzilai dan Borwein, Alternatif Minimisasi dan metode Yuan dengan kasus fungsi kuadratik ditinjau dari iterasi dan *running time*. Rata-rata hasil perbandingan menunjukkan bahwa modifikasi dengan ukuran langkah baru ini memberikan hasil yang baik untuk dimensi yang kecil dan mampu menyaingi hasil metode Barzilai-Borwein dan metode Alternatif Minimisasi untuk dimensi yang besar. Ukuran langkah baru ini memiliki kekonvergenan yang lebih cepat dibandingkan dengan metode lainnya khususnya untuk kasus dengan dimensi yang besar.

Kata Kunci: fungsi kuadratik, metode gradien, *running time, steepest descent*, ukuran langkah baru.

1 PENDAHULUAN

Pengoptimuman didefinisikan sebagai proses untuk mendapatkan keputusan terbaik yang memberikan nilai maksimum atau minimum dari suatu fungsi dengan cara penentuan solusi yang tepat. Masalah pengoptimuman dapat dikategorikan dalam dua bagian yaitu pengoptimuman berkendala dan pengoptimuman nirkendala. Pengoptimuman berkendala adalah pengoptimuman suatu fungsi dengan syarat-syarat tertentu yang membatasinya. Sebaliknya pengoptimuman nirkendala adalah pengoptimuman tanpa adanya syarat-syarat tertentu yang membatasinya [6]. Dalam menyelesaikan permasalahan pengoptimuman biasanya dapat dilakukan secara analitik maupun secara numerik.

[1]Mahasiswa S2, Program Studi Matematika Terapan, Sekolah Pascasarjana IPB Dramaga Bogor, 16680. E-mail: djihadwungguli@gmail.com
[2]Departemen Matematika, Fakultas Ilmu Pengetahuan Alam, Jalan Meranti Kampus IPB Dramaga Bogor, 16680. E-mail: bibparuhum1@yahoo.com
[3]Departemen Matematika, Fakultas Ilmu Pengetahuan Alam, Jalan Meranti Kampus IPB Dramaga Bogor, 16680. E-mail: guritman@yahoo.co.id

Untuk permasalahan nonlinear khususnya permasalahan pengoptimuman nirkendala dengan banyak variabel terdapat persoalan yang tidak mampu diselesaikan dengan metode analitik. Sehingga diperlukan metode numerik untuk menyelesaikan permasalahan tersebut. Pada umumnya, metode numerik yang digunakan dalam masalah peng-optimuman bersifat iteratif. Salah satu metode iteratif yang digunakan adalah metode *line search steepest descent*.

Metode *steepest descent* (SD) pertama kali diperkenalkan oleh Cauchy pada tahun 1847, yang merupakan salah satu prosedur paling mendasar untuk meminimumkan fungsi terdiferensialkan beberapa variabel [2]. Pada beberapa kasus, metode SD ini memiliki kekonvergenan yang lambat dalam menuju ke solusi optimum, hal ini terjadi karena langkahnya yang berbentuk zig-zag. Dalam beberapa tahun terakhir ini menjadi lebih jelas bahwa masalah penting dalam metode *steepest descent* adalah pemilihan ukuran langkah. Penentuan ukuran langkah ini dapat mempengaruhi cepat atau lambatnya kekonvergenan ke solusi optimum. Barzilai dan Borwein [1] berusaha menyempurnakan metode ini dengan memodifikasi algoritme dan hasilnya berjalan cukup baik untuk masalah dengan dimensi yang besar. Metode ini kemudian dikenal dengan metode Barzilai dan Borwein (BB).

Hasil metode BB telah memicu banyak penelitian pada metode *steepest descent*. Penelitian-penelitian yang dilakukan untuk mendapatkan algoritme pencarian ukuran langkah yang memungkinkan konvergensi cepat dan monoton. Penelitian terkait juga dilakukan oleh Dai dan Yuan [4] yang dinamakan Alternatif Minimisasi (AM) dengan ide penggabungan ukuran langkah yang bergantian antara meminimumkan nilai fungsi dan *norm* gradien disepanjang garis *steepest descent*. Penelitian lainnya dilakukan oleh Yuan [8], dengan algoritme ukuran langkah baru pada iterasi genap dan *exact line search* pada iterasi ganjil. Berdasarkan penelitian-penelitian yang telah dilakukan, maka tujuan dalam penelitian ini akan memodifikasi ukuran langkah pada metode SD dengan ukuran langkah yang baru. Selanjutnya hasil modifikasi algoritme akan dibandingkan dengan metode SD, BB, AM dan Yuan ditinjau dari iterasi dan *running time*. Modifikasi ukuran langkah yang dilakukan diharapkan dapat memperkecil iterasi dan *running time* yang dibutuhkan dari algoritme.

2 METODE *STEEPEST DESCENT*

2.1 *Steepest Descent* Klasik

Metode *steepest descent* yang diperkenalkan oleh Cauchy [3], adalah metode gradien sederhana untuk pengoptimuman nirkendala:

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}), \tag{1}$$

dimana $f(\mathbf{x})$ adalah fungsi terdiferensial kontinu di $\mathbb{R}^n$. Metode ini memiliki bentuk sebagai berikut:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k(-\mathbf{g}_k), \tag{2}$$

dimana $\mathbf{g}_k = \mathbf{g}(\mathbf{x}_k) = \nabla f(\mathbf{x}_k)$ adalah vektor gradien dari $f(\mathbf{x})$ pada saat iterasi di titik $\mathbf{x}_k$ dan $\alpha_k > 0$ adalah ukuran langkahnya [7]. Ukuran langkah α_k dapat diperoleh dengan *exact line search* :

$$\alpha_k = \text{argmin}_{\alpha>0}\{f(\mathbf{x}_k + \alpha(-\mathbf{g}_k))\}. \quad (3)$$

Pencarian ukuran langkah dengan *exact line search* (3) ini dapat menyebabkan kekonvergenan yang lambat dalam menuju ke solusi optimum, hal ini terjadi karena langkahnya yang berbentuk zig-zag. Pada kasus pengoptimuman dengan fungsi kuadratik, pencarian α_k dengan *exact line search* dapat disederhanakan menjadi

$$\alpha_k^{SD} = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_k^T A \mathbf{g}_k} \quad (4)$$

dimana *A* adalah matriks Hessian.

Algoritme 2.1 ***Steepest Descent***

Step 0. Diberikan titik awal $\mathbf{x}_0 \in \mathbb{R}^n$ dan batas toleransi $0 < \varepsilon < 1$.
Tetapkan $k = 0$.

Step 1. Tentukan $\mathbf{g_k}$. Jika $\|\mathbf{g_k}\| \leq \varepsilon$, berhenti.

Step 2. Tentukan α_k yang meminimumkan

$$f(\mathbf{x}_k + \alpha_k \mathbf{d_k}) = \min_{\alpha>0} f(\mathbf{x}_k + \alpha \mathbf{d_k})$$

atau

$$\alpha_k^{SD} = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_k^T A \mathbf{g}_k}$$

Step 3. Hitung $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g_k}$

Step 4. Beri nilai $k = k + 1$, dan pergi ke Step 1.

Penentuan ukuran langkah dalam metode *steepest descent* inilah yang menjadi pokok permasalahan. Penentuan ukuran langkah ini dapat mempengaruhi cepat atau lambatnya kekonvergenan ke solusi optimum. Adapun ukuran langkah yang telah dilakukan dalam penelitian-penelitian sebelumnya dijelaskan selanjutnya.

2.2 Barzilai dan Borwein (BB)

Metode gradien Barzilai dan Borwein [1] merupakan metode pengembangan dari metode s*teepest descent y*aitu dengan mengganti ukuran langkah α_k. Ide utama dari pendekatan metode BB adalah menggunakan informasi dalam iterasi sebelumnya untuk menentukan ukuran langkah dalam iterasi selanjutnya. Ukuran langkah metode BB diturunkan dari dua titik pendekatan ke garis potong persamaan yang didasari oleh metode quasi-Newton yaitu:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - B_k^{-1} \mathbf{g}_k \quad (5)$$

dimana $B_k^{-1} = \alpha_k I$ dan I adalah matriks identitas. Dengan ekspansi deret Taylor untuk pendekatan kuadratik yang terturunkan maka nilai B_k dapat ditentukan

dengan:

$$B_k = \arg\min_{B\in\mathbb{R}} \|B\boldsymbol{s_{k-1}} - \boldsymbol{y_{k-1}}\|^2 \tag{6}$$

atau

$$B_k = \arg\min_{B^{-1}\in\mathbb{R}} \|\boldsymbol{s_{k-1}} - B^{-1}\boldsymbol{y_{k-1}}\|^2 \tag{7}$$

dimana $\boldsymbol{s_{k-1}} = \mathbf{x_k} - \mathbf{x_{k-1}}$, dan $\boldsymbol{y_{k-1}} = \mathbf{g}_k - \mathbf{g}_{k-1}$. Dari $B_k^{-1} = \alpha_k I$ dan hubungan (6) dan (7) dapat diperoleh dua ukuran langkah yaitu:

$$\alpha_k^{BB1} = \frac{\mathbf{s}_{k-1}^T \boldsymbol{s_{k-1}}}{\mathbf{s}_{k-1}^T \boldsymbol{y_{k-1}}}, \tag{8}$$

dan

$$\alpha_k^{BB2} = \frac{\mathbf{s}_{k-1}^T \boldsymbol{y_{k-1}}}{\mathbf{y}_{k-1}^T \boldsymbol{y_{k-1}}}. \tag{9}$$

Metode BB ini sangat baik untuk kasus dengan dimensi besar akan tetapi metode BB ini tidak monoton [5]. Dalam algoritme metode BB memerlukan dua nilai $\mathbf{x_k}$ dan dua nilai $\mathbf{g}_k$, sehingga masih memerlukan *exact line search* (4) pada iterasi awal.

Algoritme 2.2 Barzilai dan Borwein (BB)

Step 0. Diberikan titik awal $\mathbf{x}_0 \in \mathbb{R}^n$ dan batas toleransi $0 < \varepsilon < 1$. Tetapkan $k = 0$.

Step 1. Tentukan $\mathbf{g_k}$. Jika $\|\mathbf{g_k}\| \leq \varepsilon$, berhenti.

Step 2. Jika k = 0 maka tentukan α_0 dengan *exact line search.* Jika tidak Tentukan α_k dengan

$$\alpha_k = \frac{\mathbf{s}_{k-1}^T \boldsymbol{s_{k-1}}}{\mathbf{s}_{k-1}^T \boldsymbol{y_{k-1}}}, \qquad \text{atau} \qquad \alpha_k = \frac{\mathbf{s}_{k-1}^T \boldsymbol{y_{k-1}}}{\mathbf{y}_{k-1}^T \boldsymbol{y_{k-1}}}$$

dimana $\boldsymbol{s_{k-1}} = \mathbf{x_k} - \mathbf{x_{k-1}}$, dan $\boldsymbol{y_{k-1}} = \mathbf{g}_k - \mathbf{g}_{k-1}$

Step 3. Hitung $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g_k}$

Step 4. Beri nilai $k = k + 1$, dan pergi ke Step 1.

2.3 Alternatif Minimisasi (AM)

Dalam beberapa pengertian, prinsipnya bahwa meminimumkan suatu fungsi $f(\mathbf{x})$ yang kontinu dan terdiferensialkan dua kali (fungsi *smooth*) adalah setara dengan meminimumkan *norm* gradien $\|\mathbf{g}(\mathbf{x})\|$. Hal ini merupakan ide dasar dari metode gradien Alternatif Minimisasi [4]. Metode gradien Alternatif Minimisasi adalah modifikasi metode *steepest descent* dengan ukuran langkah yang bergantian antara meminimumkan nilai fungsi dan *norm* gradien disepanjang garis *steepest descent*. Lebih tepatnya untuk $k \geq 1$, dipilih ukuran langkah sehingga

$$\alpha_{2k-1} = \text{argmin}_\alpha\{\|\mathbf{g}(\mathbf{x}_{2k-1} - \alpha\mathbf{g}_{2k-1})\|\} \tag{10}$$

dan

$$\alpha_{2k} = \text{argmin}_\alpha\{f(\mathbf{x}_{2k} - \alpha\mathbf{g}_{2k})\} \tag{11}$$

Sehingga dari hubungan (10) dan (11) dapat diperoleh

$$\alpha_k^{AM} = \begin{cases} \dfrac{\mathbf{g}_k^T A \mathbf{g_k}}{\mathbf{g}_k^T A^2 \mathbf{g_k}}, & \text{jika } k \text{ ganjil} \\ \dfrac{\mathbf{g}_k^T \mathbf{g_k}}{\mathbf{g}_k^T A \mathbf{g_k}}, & \text{jika } k \text{ genap} \end{cases} \tag{12}$$

Algoritme 2.3 Alternatif Minimisasi (AM)

Step 0. Diberikan titik awal $\mathbf{x}_0 \in \mathbb{R}^n$ dan batas toleransi $0 < \varepsilon < 1$.
Tetapkan $k = 1$.

Step 1. Tentukan $\mathbf{g_k}$ dan A. Jika $\|\mathbf{g_k}\| \leq \varepsilon$, berhenti.

Step 2. Jika k ganjil maka tentukan

$$\alpha_k = \frac{\mathbf{g}_k^T A \mathbf{g_k}}{\mathbf{g}_k^T A^2 \mathbf{g_k}}$$

Jika tidak tentukan

$$\alpha_k = \frac{\mathbf{g}_k^T \mathbf{g_k}}{\mathbf{g}_k^T A \mathbf{g_k}}$$

Step 3. Hitung $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g_k}$

Step 4. Beri nilai $k = k + 1$, dan pergi ke Step 1.

2.4 Metode Yuan

Metode gradien Yuan [8] menggunakan ukuran langkah yang bergantian seperti yang dilakukan pada metode AM. Akan tetapi metode Yuan menggunakan ukuran langkah yang baru. Ukuran langkah Yuan menggunakan *exact line search* (4) pada iterasi ganjil, kemudian menggunakan ukuran langkah Yuan pada iterasi genap yaitu:

$$a_{2k}^Y = \frac{2}{\sqrt{(1/\alpha_{2k-1} - 1/\alpha_{2k})^2 + 4\|\mathbf{g}_{2k}\|^2/\|\mathbf{s}_{2k-1}\|^2} + 1/\alpha_{2k-1} + 1/\alpha_{2k}} \tag{13}$$

dimana $\mathbf{s}_{2k-1} = \mathbf{x}_{2k} - \mathbf{x}_{2k-1} = -\alpha_{2k-1}\mathbf{g}_{2k-1}$. Secara umum pencarian ukuran langkah α_k pada metode Yuan dituliskan sebagai berikut:

$$\alpha_k = \begin{cases} \alpha_k^{SD}, & \text{jika } k \text{ ganjil} \\ a_k^Y, & \text{jika } k \text{ genap.} \end{cases} \tag{14}$$

Algoritme 2.4 Metode Yuan

Step 0. Diberikan titik awal $\mathbf{x}_1 \in \mathbb{R}^n$ dan batas toleransi $0 < \varepsilon < 1$.

Step 1. Tentukan $\mathbf{g_1}$ dan A. Jika $\|\mathbf{g_1}\| \leq \varepsilon$, berhenti. Tetapkan $k = 1$.

Step 2. Tentukan α_{2k-1}. Kemudian hitung

$\mathbf{x}_{2k} = \mathbf{x}_{2k-1} - \alpha_{2k-1}\mathbf{g}_{2k-1}$

Step 3. Jika $\|\mathbf{g_{2k}}\| \leq \varepsilon$, berhenti

Step 4. Tentukan α_{2k} dan $\mathbf{s}_{2k-1} = \mathbf{x}_{2k} - \mathbf{x}_{2k-1}$. Tentukan

$$a_{2k}^{Y} = \frac{2}{\sqrt{(1/\alpha_{2k-1} - 1/\alpha_{2k})^2 + 4\|\mathbf{g}_{2k}\|^2/\|\mathbf{s}_{2k-1}\|^2} + 1/\alpha_{2k-1} + 1/\alpha_{2k}}$$

Hitung $\mathbf{x}_{2k+1} = \mathbf{x}_{2k} - a_{2k}^{Y}\mathbf{g}_{\mathbf{2k}}$

Step 5. Jika $\|\mathbf{g}_{\mathbf{2k+1}}\| \leq \varepsilon$, berhenti

Step 6. Beri nilai $k = k + 1$, dan pergi ke Step 2.

3 MODIFIKASI *STEEPEST DESCENT*

Telah dibahas pada bagian sebelumnya bahwa pada [8] menggunakan ukuran langkah dengan *exact line search* pada iterasi ganjil dan kemudian menggunakan ukuran langkah (13) pada iterasi genap yang secara umum dituliskan seperti pada (14). Berdasarkan metode Yuan ini maka akan dilakukan modifikasi ukuran langkah dengan ukuran langkah yang baru. Untuk ukuran langkah baru ini dapat dibentuk dari suatu algoritme dengan langkah sebagai berikut:

$$\begin{aligned} \mathbf{x}_2 &= \mathbf{x}_1 - \alpha_1\mathbf{g}_1 \\ \mathbf{x}_3 &= \mathbf{x}_2 - \alpha_2^{Y}\mathbf{g}_2 \\ \mathbf{x}_4 &= \mathbf{x}_3 - \alpha_3^{Y}\mathbf{g}_3 \\ \mathbf{x}_5 &= \mathbf{x}_4 - \alpha_4\mathbf{g}_4 \end{aligned} \tag{15}$$

dimana α_1^{SD} dan α_2^{SD} merupakan ukuran langkah dengan proses pencarian menggunakan *exact line search* (4), sedangkan untuk α_3^{Y} dan α_4^{Y} menggunakan ukuran langkah Yuan (13). Algoritme (15) ini merupakan bentuk awal dari proses iterasi, sehingga untuk proses iterasi (15) selanjutnya akan terus berlanjut sampai solusi nilai $\mathbf{x}$ ditemukan. Secara umum ukuran langkah α_k dari bentuk (15) dapat dituliskan sebagai berikut:

$$\alpha_k = \begin{cases} \alpha_k^{SD}, & \text{jika } mod(k,4) = 1 \text{ atau } 2 \\ a_k^{Y}\ , & \text{jika } k \text{ genap} \end{cases} \tag{16}$$

Algoritme 3.1 Ukuran Langkah Baru (a)

Step 0. Diberikan titik awal $\mathbf{x}_0 \in \mathbb{R}^n$ dan batas toleransi $0 < \varepsilon < 1$.
Tetapkan $k = 1$.

Step 1. Tentukan $\mathbf{g}_\mathbf{k}$ dan *A*. Jika $\|\mathbf{g}_\mathbf{k}\| \leq \varepsilon$, berhenti.

Step 2. Jika *mod*(*k*,4) = 1 atau 2 maka tentukan

$$\alpha_k = \frac{\mathbf{g}_k^T\mathbf{g}_\mathbf{k}}{\mathbf{g}_k^T A\mathbf{g}_\mathbf{k}}$$

Jika tidak tentukan $\mathbf{s}_{k-1} = \mathbf{x}_k - \mathbf{x}_{k-1}$ dan

$$a_k^{Y} = \frac{2}{\sqrt{(1/\alpha_{k-1} - 1/\alpha_k)^2 + 4\|\mathbf{g}_k\|^2/\|\mathbf{s}_{k-1}\|^2} + 1/\alpha_{k-1} + 1/\alpha_k}$$

Step 3. Hitung $\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k\mathbf{g}_\mathbf{k}$

Step 4. Beri nilai $k = k + 1$, dan pergi ke Step 1.

Tabel 1
Rata-rata jumlah iterasi dari metode *steepest descent* untuk dimensi 2 dan 3

n	λ_n	Jumlah iterasi					
		SD	BB1	BB2	AM	Yuan	(3.1)
2	10	15.8	9.4	8.8	7.8	3.0	5.0
	100	17.6	10.2	9.4	8.0	3.0	5.0
	1000	18.0	9.6	8.2	6.6	3.0	5.0
3	10	58.0	16.0	15.4	21.6	15.0	14.8
	100	446.0	19.4	15.0	41.2	9.4	11.4
	1000	**	25.2	14.8	63.2	7.0	10.6

** Lebih dari 2000 iterasi

Tabel 2
Rata-rata *running time* dari metode *steepest descent* untuk dimensi 2 dan 3

n	λ_n	*Running time* (*s*)					
		SD	BB1	BB2	AM	Yuan	(3.1)
2	10	1.2158	0.7921	0.7430	0.6681	0.3445	0.4813
	100	1.3592	0.8915	0.7654	0.7447	0.3729	0.5298
	1000	1.2957	0.7972	0.6808	0.5900	0.3510	0.4789
3	10	4.1901	1.5578	1.2570	1.7168	1.2100	1.1979
	100	38.2374	1.6251	1.1804	3.0742	0.8563	0.9336
	1000	**	2.3408	1.2184	4.7670	0.6885	0.9288

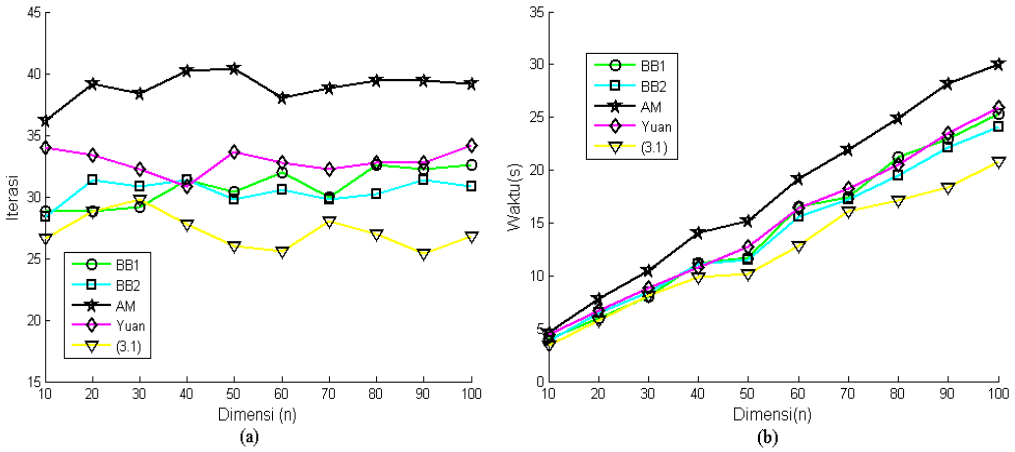
** Lebih dari 1200 sekon

4 HASIL NUMERIK

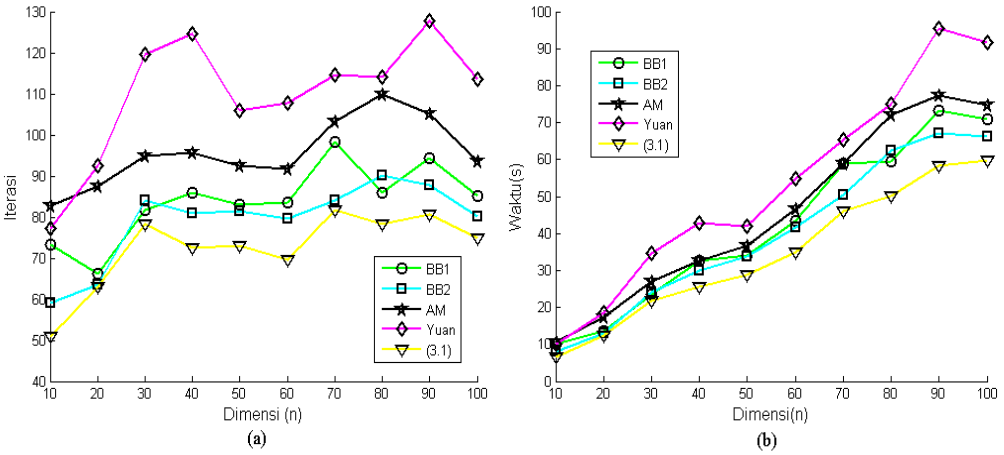
Pada bagian ini dilakukan perbandingan hasil numerik dari setiap metode *steepest descent* yang telah dijelaskan pada bagian sebelumnya, yaitu SD, BB, AM, Yuan, dan algoritme (3.1). Ukuran langkah untuk metode SD menggunakan (4) yang merupakan penyederhanaan dari *exact line search*. Untuk metode BB dibagi menjadi dua bagian, yaitu BB1 menggunakan ukuran langkah (8) dan BB2 menggunakan ukuran langkah (9). Adapun hal yang dibandingkan dalam penelitian ini adalah jumlah iterasi dan *running time* yang diperlukan dari setiap metode untuk mendapatkan nilai minimum. Perbandingan dilakukan untuk kasus fungsi nonlinear dengan bentuk kuadratik. Fungsi kuadratik yang digunakan dalam penelitian ini dibangkitkan secara acak dalam bentuk

$$f(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mathbf{x}^*)^T Diag(\lambda_1, \dots, \lambda_n)(\mathbf{x}-\mathbf{x}^*), \quad \mathbf{x} \in \mathbb{R}^n,$$

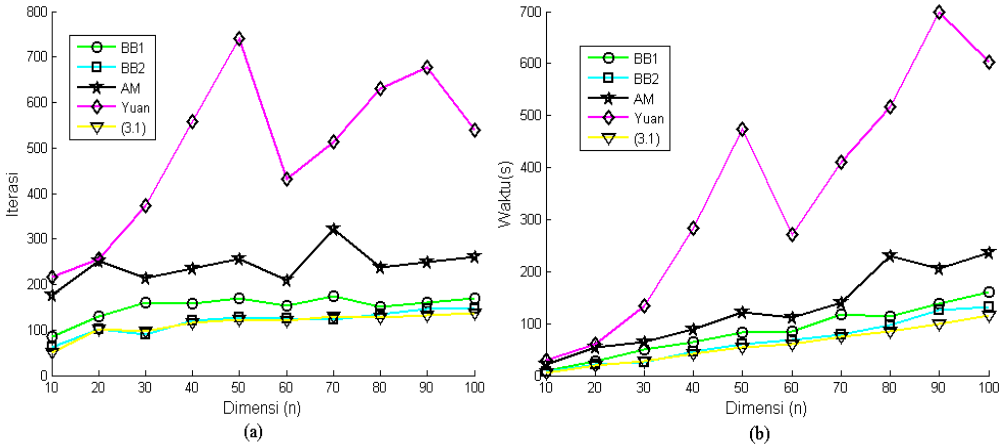
n menyatakan dimensi dari fungsi kuadratik dengan $n = 2, 3, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100$. Vektor $\mathbf{x}_i^*(i = 1, \cdots, n)$ merupakaan bilangan acak integer pada interval $[-5,5]$. Selanjutnya $\lambda_1 = 1$ dan $\lambda_n = 10,100,1000$ merupakan kondisi dari matriks Hessian dari fungsi. Kemudian $\lambda_i(i = 2, \cdots, n-1)$ adalah bilangan acak integer pada interval $[1, \lambda_n]$. Untuk semua dimensi dan λ_n diberikan titik awal vektor nol $(0, \cdots 0)^T$ dan kriteria penghentian adalah $\|\mathbf{g}_k\| \leq 10^{-8}$. Percobaan dilakukan sebanyak 5 kali untuk setiap dimensi dan setiap λ_n dari setiap metode. Sehingga percobaan yang dilakukan untuk satu dimensi sebanyak 90 kali dan total percobaan untuk semua dimensi sebanyak 1080 kali.



Gambar 1 Perbandingan rata-rata jumlah iterasi (a) dan *running time* (b) dari metode *steepest descent* untuk λ_n = 10



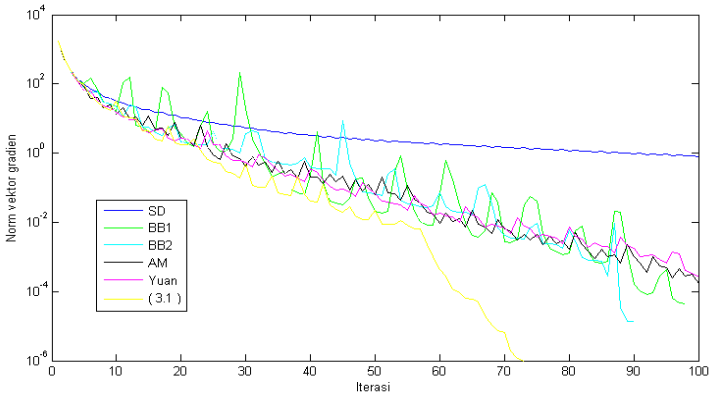
Gambar 2 Perbandingan rata-rata jumlah iterasi (a) dan *running time* (b) dari metode *steepest descent* untuk λ_n = 100



Gambar 3 Perbandingan rata-rata jumlah iterasi (a) dan *running time* (b) dari metode *steepest descent* untuk λ_n = 1000

Berdasarkan hasil percobaan yang dilakukan secara umum dapat dilihat hubungan antara dimensi dan λ_n dengan iterasi dan *running time*. Untuk kasus dimensi kecil (n = 2 & 3) dapat dilihat bahwa besarnya λ_n tidak berpengaruh terhadap iterasi dan *running time*. Artinya semakin besar λ_n tidak menjamin bahwa iterasi dan *running time* akan semakin besar pula (Tabel 1 dan 2). Untuk kasus dimensi besar (n = 10,20,⋯,100) terlihat bahwa semakin besar λ_n maka semakin besar pula iterasi dan *running time*nya (Gambar 1 – 3). Selanjutnya dapat dilihat bahwa semakin besar dimensi tidak menyebabkan semakin besar iterasinya (Gambar 1a – 3a). Berbeda halnya dengan *running time*, semakin besar dimensi maka semakin besar pula *running time*nya (Gambar 1b – 3b).

Pada masalah fungsi kuadratik dengan dimensi yang kecil untuk semua λ_n (10,100,1000), dapat dilihat bahwa metode Yuan mampu menemukan solusi nilai minimum dengan iterasi dan *running time* yang terkecil. Meskipun demikian bahwa algoritme (3.1) mampu menyeimbangi kecepatan metode Yuan untuk dimensi yang kecil. Untuk kasus dengan dimensi besar, metode Yuan memberikan hasil yang buruk khususnya pada $\lambda_n = 100$ dan $\lambda_n = 1000$. Berbeda dengan algoritme (3.1), untuk kasus dimensi besar dengan semua ukuran λ_n, algoritme (3.1) lebih dominan menemukan solusi nilai minimum fungsi dengan iterasi dan *running time* yang terkecil dibandingkan dari metode BB, AM, dan Yuan (Gambar 1 – 3). Hal ini terjadi karena algoritme (3.1) memiliki tingkat konvergensi yang lebih cepat dibandingkan dengan metode gradien lainnya, sebagai contoh ditunjukkan pada gambar 4 dengan n = 100 dan λ_n = 100. Meskipun demikian pada dimensi dan λ_n tertentu metode BB dapat mengungguli algoritme (3.1).



Gambar 4 Kekonvergenan dari metode *steepest descent*

5 SIMPULAN

Metode Yuan merupakan metode gradien yang mampu memberikan iterasi dan *running time* yang kecil untuk kasus fungsi kuadratik dengan dimensi yang kecil. Akan tetapi metode Yuan memberikan kinerja yang buruk untuk kasus dengan dimensi dan λ_n yang lebih besar. Dalam penelitian ini dilakukan modifikasi ukuran langkah metode *steepest dengan* modifikasi ukuran langkah pada metode Yuan. Berbeda dengan metode Yuan, modifikasi ukuran langkah baru ini memberikan kinerja yang lebih baik untuk kasus fungsi kuadratik dengan dimensi dan λ_n yang besar. Bahkan modifikasi ukuran langkah baru ini lebih dominan mampu mengungguli kinerja dari metode Barzilai dan Borwein dan metode Alternatif Minimisasi. Hal ini terlihat dari rata-rata iterasi dan *running time* hasil percobaan yang dilakukan dengan fungsi kuadratik yang dibangkitkan secara acak. Modifikasi ukuran langkah baru ini mempunyai tingkat konvergensi yang cepat dibandingkan metode gradien yang lainnya.

DAFTAR PUSTAKA

[1] Barzilai J, Borwein JM. 1988. Two point step size gradient methods. *IMA Journal of Numerical Analysis*, 8: 141-148.

[2] Bazara MS, Sherali HD, Shetty CM. 2006. *Nonlinear Programming: Theory and Algorithms*. USA: Wiley-Interescience.

[3] Cauchy A. 1847. General method for solving simultaneous equations Systems, *Comp. Rend. Sci*. *Paris,* 25: 46-89.

[4] Dai YH, Yuan Y. 2003. Alternate minimization gradient method. *IMA Journal of Numerical Analysis*, 23: 377-393.

[5] Dai YH. 2003. A New Analysis on the Barzilai-Borwein Gradient Method. Operations Research Society of China, Periodicals Agency of Shanghai University and Spinger –Verlag Berlin Heidelberg.

[6] Griva I, Nash SG, Sofer A. 2009. *Linear and Nonlinear Optimization*. USA: Society for Industrial and Applied Mathematics.

[7] Sun Wenyu, Yuan Y. 2006. *Optimization Theory and Methods. Nonlinear Programming*. New York: Spinger Science, Business Media.

[8] Yuan Y. 2006. A new stepsize for the steepest descent method. *Journal of Computational Mathematics*, 24:149-156.