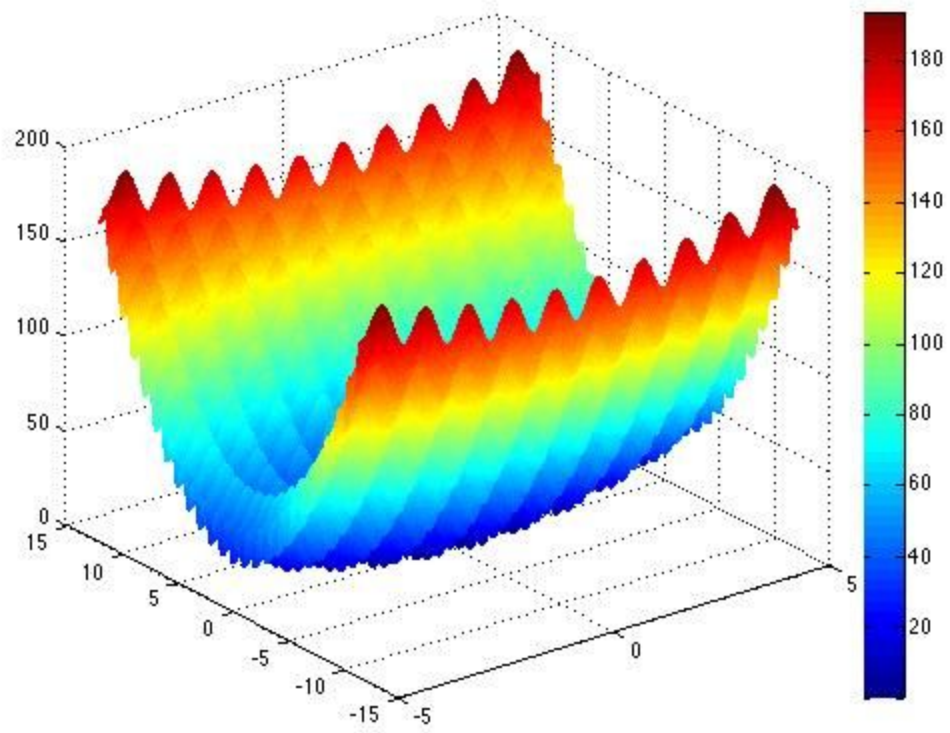
Trabalho Computacional 1
Computação Evolucionária

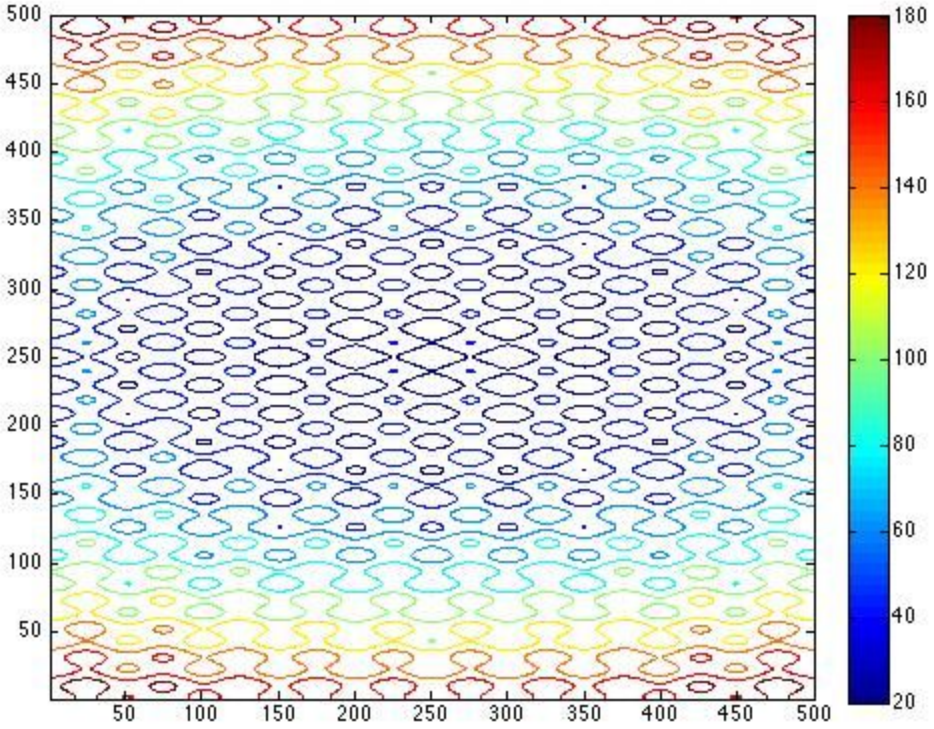
Aluno: Tarcísio Bruno C. Oliveira
Matrícula: 382084

1. Gráfico da função f(x1, x2) para todo o dominío x1 e x2.



Podemos observar pelo gráfico que essa função possui um mínimo global resultante dos termos quadráticos e inúmeros máximos e mínimos locais resultantes do efeito das funções cossenos.

2. Gráfico das curvas de contorno para f(x1, x2).



No gráfico das curvas de níveis acima fica evidente a presença de máximos e mínimos locais representados pelos círculos na figura.

3. Encontrar o mínimo global usando GA tomando como base o código Matlab/Octave enviado por email. Mostrar gráficos da função de aptidão do melhor indivíduo e da aptidão média da população a cada geração. Especificar os valores adequados dos parâmetros tamanho da população (N), e probabilidades de recombinação (pc) e de mutação.

3.1 Escolha dos parâmetros:

a. Tamanho do cromossomo: 40

Eu defini o tamanho do cromossomo para que eu tenha uma precisão na 5ª casa decimal para x1 e na 4ª casa decimal para x2.

Assim, o número de bits que eu escolhi para representar cada variável foi de 20 bits, ou seja, um cromossomo de 40 bits

b. Tamanho da população: 100

O tamanho da população que eu escolhi foi 100. Eu rodei o algoritmo com tamanho da população 50 e 100. Escolhi 100 para ter uma maior número de amostras e ter um gráfico da média mais suavizado.

c. Número de gerações: 50

Eu rodei primeiramente com 100 gerações mas notei que o valor tende a convergir entre 20 e 30 gerações, então deixei 50 gerações

d. Probabilidade de Recombinação (0.95) e Mutação (0.01)

Deixei como estava no código do professor. Próxima questão abordarei mais em detalhe esses dois parâmetros.

3.2 Resultados

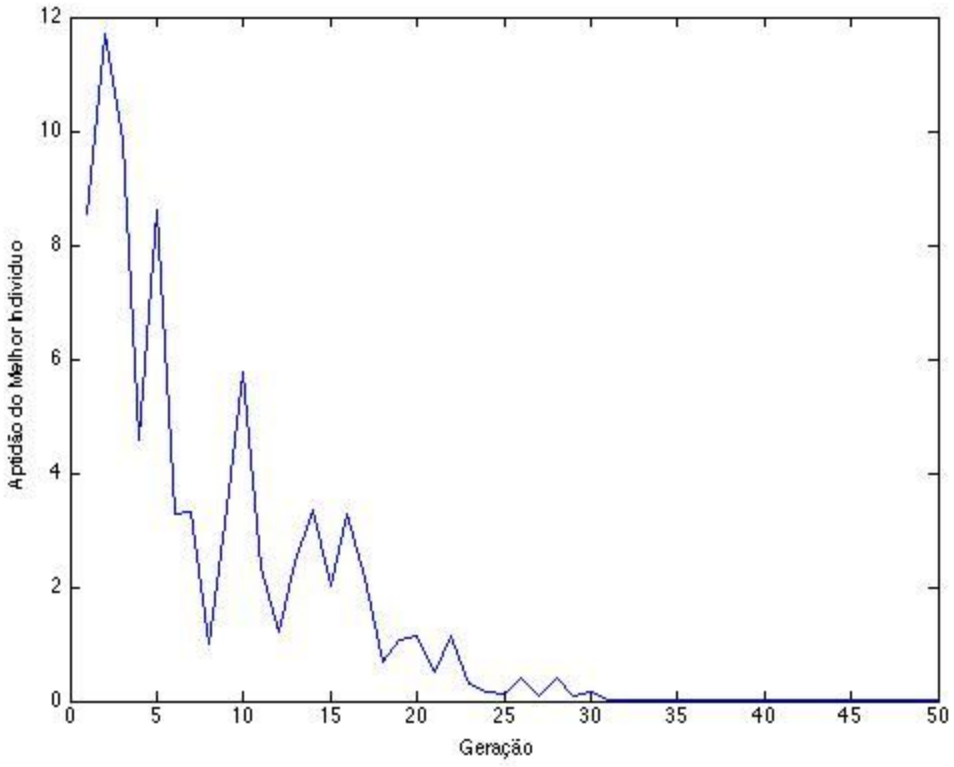
Os valores encontrados para o melhor indivíduo da última geração de uma execução do algoritmo.

X1 = -0.0021

X2 = 0.0015

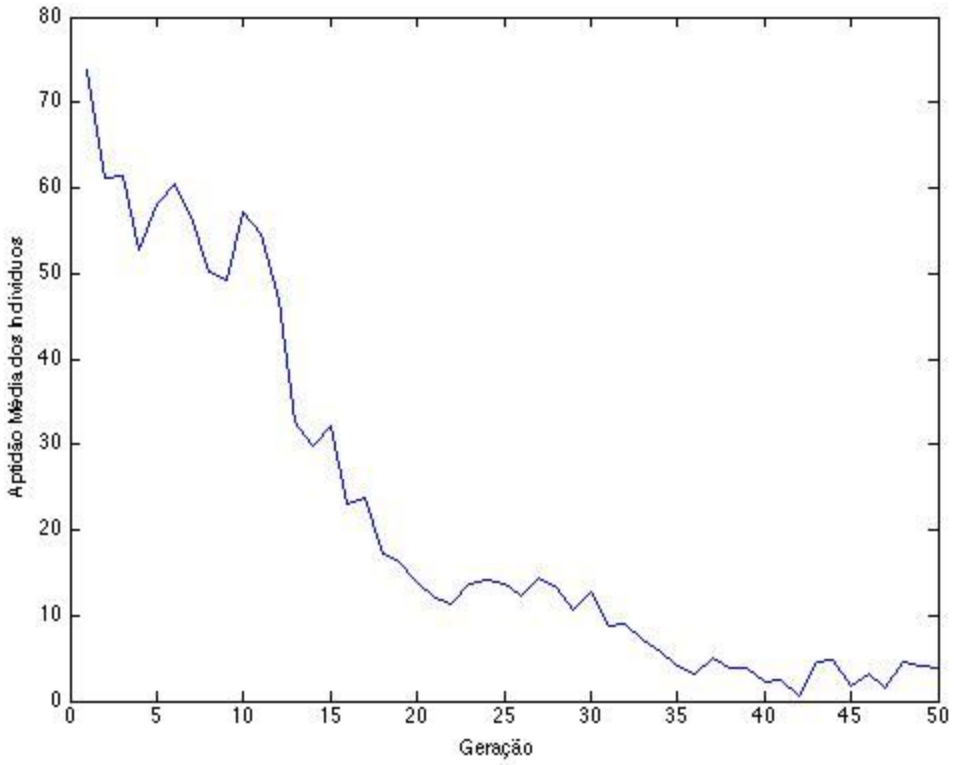
Fmin = 4.4433e-06

3.2.1 Gráfico da função de aptidão do melhor indivíduo a cada geração



Podemos observar que o algoritmo convergiu logo após a 30ª geração para um valor bem próximo a zero.

3.2.2 Gráfico da aptidão média da população a cada geração



Novamente verificamos que o algoritmo converge após a 30ª geração e a curva da média sofre flutuações suaves.

4. Avaliar empiricamente o efeito de uma escolha inadequada dos parâmetros (N, pc e pm) no desempenho do AG.

Para essa questão irei fixar o N em 100 como utilizado na questão anterior e irei variar o valor de pc e pm.

4.1 Variando o valor de pc.

Ao variar o valor de pc entre 0.95 e 0.01 notei que não houve muito diferença nos resultados. Uma das diferenças é que o resultado converge mais rápido, no entanto, ele não converge para um valor tão próximo do esperado. Isso acontece pois coloquei um valor de N muito alto.

4.2 Variando o valor de pm

Ao variar o valor de pm entre 0.9 e 0.01 percebi que valores mais altos de pm fazem o algoritmo não convergir para o valor esperado. O tamanho do N tem impacto nessa influência. O script com N mais alto sofre menos influência.

4.3 Conclusão

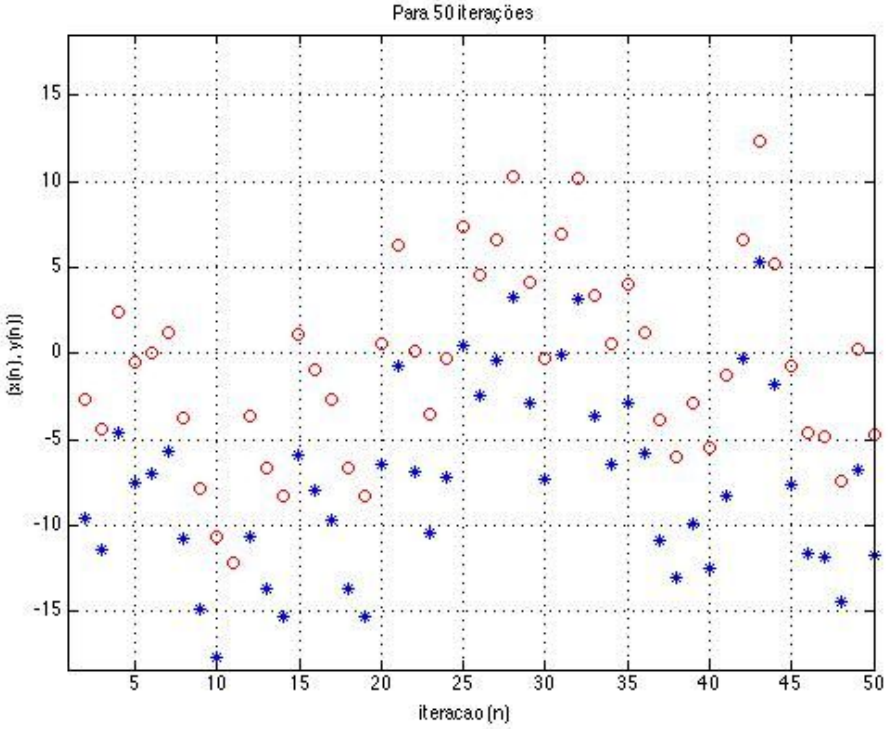
Podemos observar que o algoritmo deve possuir baixos valores de probabilidade de mutação e altos valores de probabilidade de recombinação para ele convergir para um valor mais próximo do esperado.

A escolha do N tem impacto direto na sensibilidade a esses fatores. Escolhendo um N alto como eu escolhi torna o algoritmo menos sensível a esses outros parâmetros.

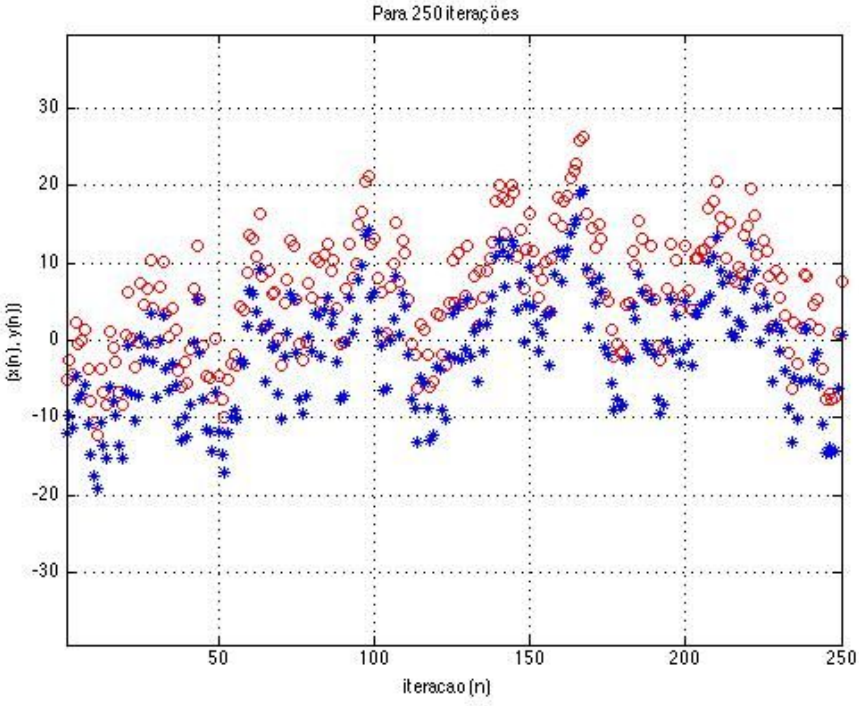
5. Repetir o experimento usando a metaheurística Hill-Climbing e o método do gradiente descendente. Compare os resultados obtidos em termos de velocidade de convergência para o ótimo global, tempo de simulação e a insensibilidade a variação de parâmetros.

5.1 Gradiente Descendente

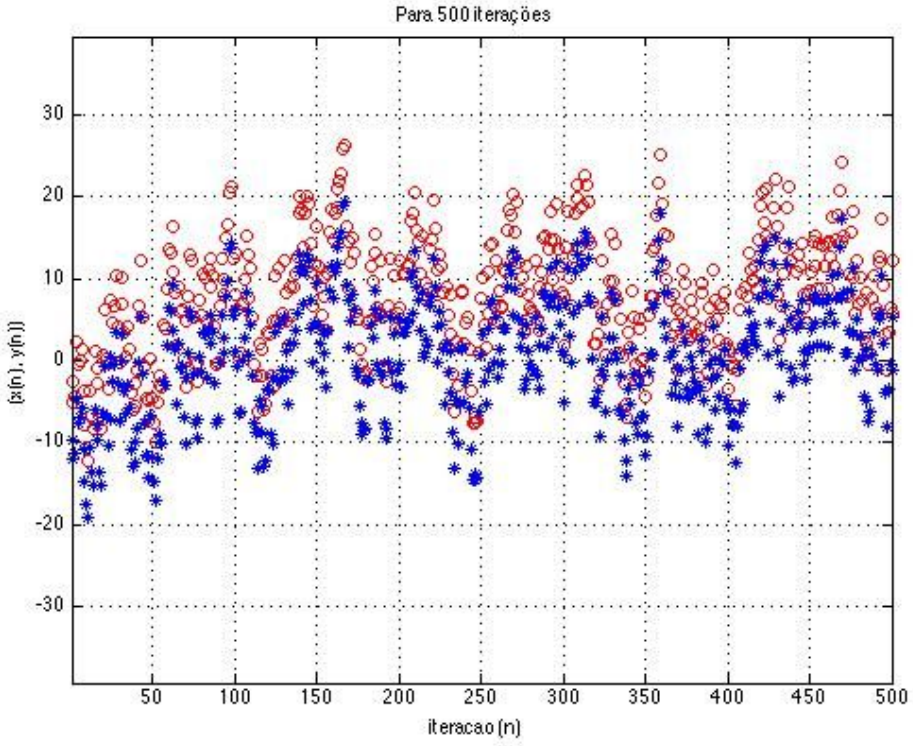
O algoritmo do gradiente descendente pelo método iterativo possui o problema de ficar preso em máximos e mínimos locais. Para a função f(x1, x2) ele não convergiu para o mínimo global. O algoritmo foi rodado para 50, 250 e 500 iterações. Confira o resultado abaixo:



Com 50 iterações



Com 250 Iterações



Com 500 Iterações

5.2 Hill Climbing vs Algoritmo Genético Binário

Nesta seção será feita uma análise de ambos algoritmos baseado em: velocidade de convergência, tempo de simulação e insensibilidade à variação de parâmetros.

5.2.1 Velocidade de Convergência

Na maioria das vezes o AG converge por volta da 30ª geração, já o Hill Climbing varia bastante quanto à convergência. Isso ocorre devido ao Hill Climbing ser sensível a ótimos locais e acaba ficando preso em um deles.

Então, algumas vezes ele converge rápido, mas não é o ótimo global. Em outras execuções ele demora mais de 100 gerações (rodei para 500 gerações). Logo, nesse quesito o AG é mais confiável.

5.2.2 Tempo de simulação

Fiz o experimento com 40 execuções de cada algoritmo.

Algoritmo	Tempo Médio (s)
Hill Climbing	0.0670 ± 0.0065 s
AG Binário	0.6045 ± 0.0189 s

O Hill Climbing executa quase 10x mais rápido que o AG Binário. Isso era esperado já que o Hill Climbing é uma simplificação do AG Binário. Além de só possuir um indivíduo, o que reduz processamento tanto na etapa de seleção quanto no cálculo da aptidão, ele também não executa recombinação, apenas mutação.

Logo, caso a função a ser estudada não possua ótimos locais (ou pelo menos tenha poucos) e o tempo de processamento seja um recurso crítico o Hill Climbing é mais indicado.

5.2.3 Insensibilidade a variação de parâmetros

Como disse anteriormente, o Hill Climbing é uma simplificação do AG Binário. Logo, ele possui menos parâmetros que o AG Binário. Basicamente, você pode alterar o número de gerações e tamanho do cromossomo. A probabilidade de mutação é 1/(Tamanho do cromossomo). Assim, o Hill Climbing é bastante sensível ao tamanho do cromossomo, enquanto o AG Binário é mais flexível em relação a que parâmetro você quer mexer.