Notes

Kopnov Alexandr

2 апреля 2023 г.

Содержание

1 Постановка

Найти

$$\min f(x) \qquad\qquad x \in \mathbb{R}^n$$

$$f(x) = x_1^2 + 4x_2^2 + \sin(6x_1 + 7x_2) + 3x_1 + 2x_2$$

$$\nabla f(x) = \frac{df}{dx_1}\vec{e}_1 + \frac{df}{dx_2}\vec{e}_2$$

$$\frac{df}{dx_1} = 2x_1 + 6\cos(6x_1 + 7x_2) + 3$$

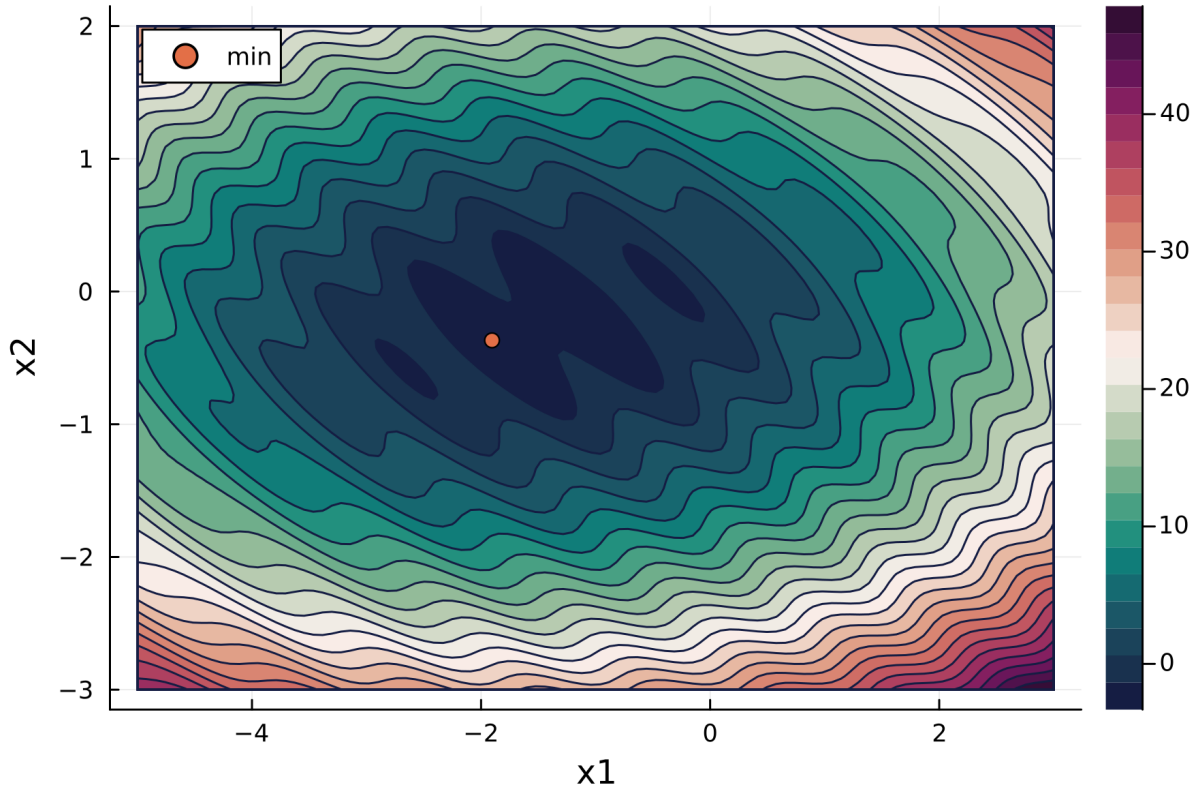
$$\frac{df}{dx_2} = 8x_2 + 7\cos(6x_1 + 7x_2) + 2$$

Строим $\{x_k\}$ — релаксационную последовательность спуска: $f(x_{k+1}) \leqslant f(x_k)$, используя двухшаговую систему: $x_{k+1} = x_k + \alpha_k p_k$.

График функции, также отмечено значение минимума, получаемое с помощью градиентного спуска в библиотеке `Optim` языка `Julia` с начальной точкой `(0,0)`

```
f(x :: Vector{Float64}) = x[1]^2 + 4*x[2]^2 + sin(6*x[1] + 7*x[2]) + 3*x[1] + 2*x[2];
x1 = range(-5,3, length=100);
x2 = range(-3,2, length=100);
vals = [f([x,y]) for y in x2, x in x1];
minim = Optim.minimizer(optimize(f,zeros(2), GradientDescent()));
# p1 = surface(x1,x2,vals, c=:curl, xlabel="x1", ylabel="x2", cam=(120,30));
# scatter!([minim[1]],[minim[2]],[f(minim)], label="min")
p2 = contour(x1,x2,vals,levels=25,fill=true, c=:curl, dpi = 300, xlabel="x1", ylabel="x2");
scatter!([minim[1]],[minim[2]], label="min");
```

```
# plot(p1,p2, size=(500,300), dpi=300)
savefig("figs/plot.png")
```



Точка минимума и значение функции в ней

```
([-1.9043886748215388, -0.36794672198200706], -3.2716974195454807)
```

Результаты работы функции оптимизации в `Julia`

```
f(x) = x[1]^2 + 4*x[2]^2 + sin(6*x[1] + 7*x[2]) + 3*x[1] + 2*x[2];

optimize(f, zeros(2), GradientDescent())
```

```
 * Status: success

 * Candidate solution
    Final objective value:     -3.271697e+00

 * Found with
    Algorithm:     Gradient Descent

 * Convergence measures
    |x - x'|               = 4.29e-09  0.0e+00
    |x - x'|/|x'|          = 2.25e-09  0.0e+00
    |f(x) - f(x')|         = 0.00e+00  0.0e+00
    |f(x) - f(x')|/|f(x')| = 0.00e+00  0.0e+00
```

```
    |g(x)|                 = 1.82e-07   1.0e-08

* Work counters
   Seconds run:   0  (vs limit Inf)
   Iterations:    167
   f(x) calls:    425
   f(x) calls:   425
```

2 Градиентный метод наискорейшего спуска

Выбираем $\epsilon > 0$, x_0, $0 < \alpha_0 < 1$ — начальное приближение.

На k-й итерации:

1. Вычисляем градиент f. Реализуется как вектор функций — частных производных.
2. Подбираем шаг: $\alpha_k \in (0, 1] : f(x_k - \alpha_k \nabla f(x_k)) = \min$.
   a) Метод золотого сечения
   b) Метод дихотомии
3. $x_{k+1} = x_k - \alpha_k \nabla f(x_k)$.

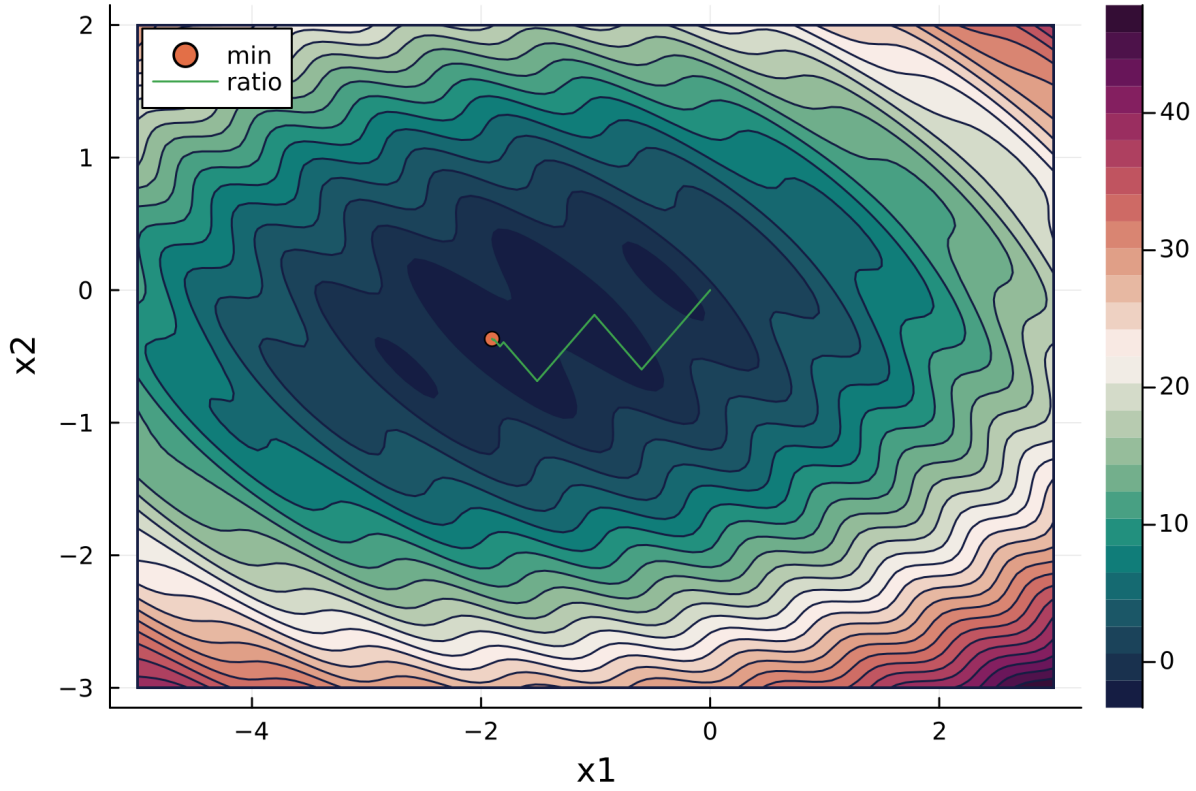
Условие выхода из цикла:

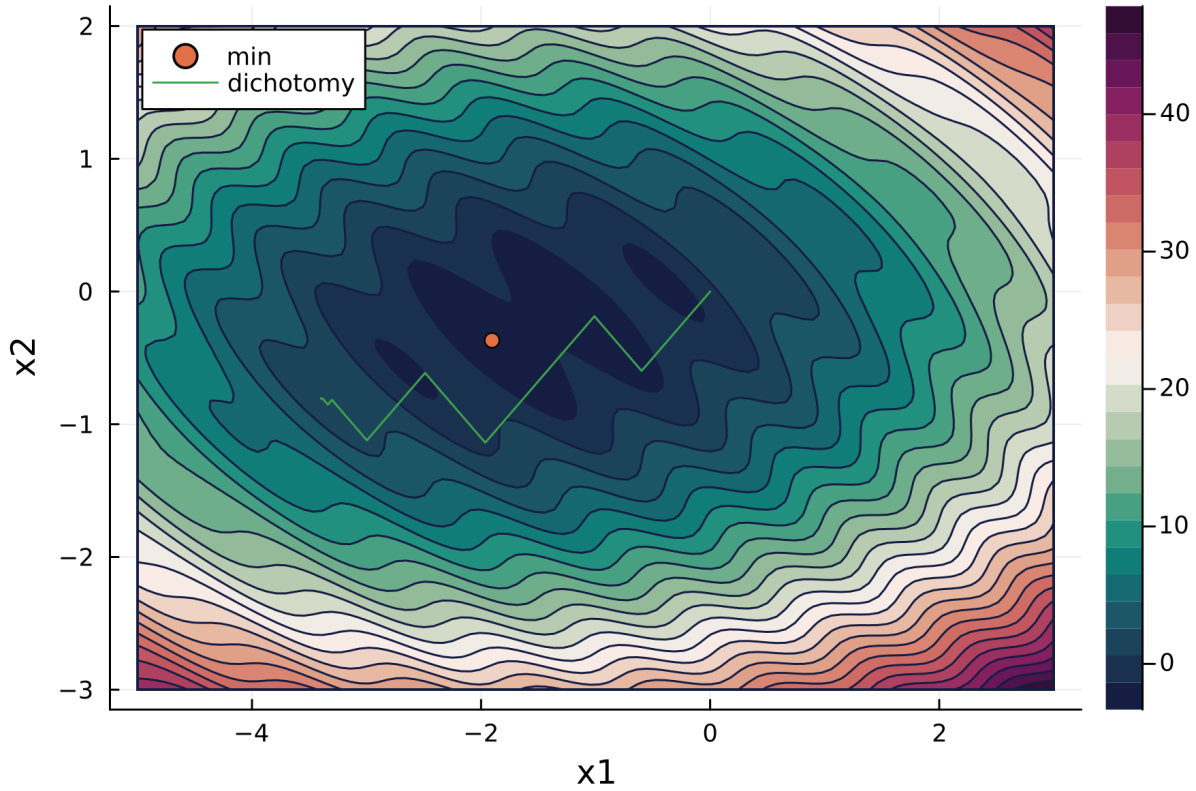
$$\|\nabla f(x_k)\| < \epsilon$$

Как правило, используют евклидову норму.

Для обоснования сходимости, используем Thm 3. О сходимости градиентного метода наискорейшего спуска, что требует доказательства липшицевости градиента.

```
df = CSV.read("build/res/" * name, DataFrame; header=["x","y"], delim=' ')
@df df plot(p2, :x, :y, labels = name)
savefig("figs/" * name * ".png")
```





3 Градиентный метод второго порядка

Выбор шага производим по принципу дробления.