Постановка задачи

Минимизировать функцию $f(x) = \frac{1}{2}x^T Ax + bx$, где $x \in \mathbb{R}^6$,

$A_{6\times 6}$ — произвольная положительно определенная матрица, A∈ {R} ^ {n×n}

b — произвольный ненулевой вектор размерности 6, $b \in \mathbb{R}^{n\times 1}$

x_0 — произвольный начальный ненулевой вектор размера 6, отдаленный от точного решения, $x \in \mathbb{R}^{n\times 1}$

$$A^* = \begin{pmatrix} 65.1140547 & -26.7627461 & -22.0076980 & -82.7767129 & 35.0062152 & 75.5819058 \\ -26.7627461 & 229.0660173 & 19.9323298 & 11.3176738 & -41.5887940 & 27.5077199 \\ -22.0076980 & 19.9323298 & 47.8558662 & 92.3011760 & -20.1879154 & -18.1501243 \\ -82.7767129 & 11.3176738 & 92.3011760 & 231.2486741 & -30.5692642 & -103.3184336 \\ 35.0062152 & -41.5887940 & -20.1879154 & -30.5692642 & 130.1539779 & 82.9497276 \\ 75.5819058 & 27.5077199 & -18.1501243 & -103.3184336 & 82.9497276 & 189.0452850 \end{pmatrix}$$

**Из-за нехватки места количество записываемых десятичных цифр сократилось до 7.*

Процедура создания матрицы A заключается в создании случайной обратимой матрицы U (с определителем больше 0) и присвоении A равного $M^T M$. Это гарантирует, что A будет положительно определенной матрицей.

Тем не менее, можно убедиться, что матрица A является положительно определенной, получив ее собственные значения и проверив, все ли они положительны и что A симметрична. Ниже приводится набор собственных значений A:

$$\lambda = \{1.172, 17.858, 60.389, 158.527, 245.585, 408.953\}$$

$$b = \begin{pmatrix} 9.814255243856422 \\ -0.3400832138148502 \\ -4.2742798826246275 \\ 0.5729798470675629 \\ 1.0452978781511018 \\ -6.258477409122025 \end{pmatrix} \qquad x_0 = \begin{pmatrix} -6.998099039133139 \\ -5.532602553195874 \\ 5.222094920523634 \\ 3.71860843139695 \\ -5.88821505753657 \\ 4.175398532986163 \end{pmatrix}$$

$$x_{\text{точ}} = -A^{-1}b = \begin{pmatrix} -1.550648425654797 \\ -0.22570089760570633 \\ 3.4869725406369905 \\ -2.0145864008544634 \\ 0.6327463127507499 \\ -0.3579729727467522 \end{pmatrix}$$

Метод градиента

$x_{k+1} = x_k - \lambda f'(x_k)$, где $\lambda = 10^{-4}$

Первая производная функции: $f'(x) = \frac{1}{2}(A^T + A)x + b$

Приравнивая производную к нулю, получаем вектор $x_{\text{точ}} \in \mathbb{R}^{n\times 1}$

$$x_{\text{точ}} = -A^{-1}b = \begin{pmatrix} -1.550648425654797 \\ -0.22570089760570633 \\ 3.4869725406369905 \\ -2.0145864008544634 \\ 0.6327463127507499 \\ -0.3579729727467522 \end{pmatrix}$$

Алгоритм отработал за 24406 шагов. Условие выхода из цикла: $\|x_{k+1} - x_k\| < \varepsilon = 10^{-5}$

Промежуточные результаты:

$$x_{\frac{m}{4}} = x_{6101} = \begin{pmatrix} -1.3719100792558692 \\ -0.20442898556213976 \\ 2.887074967708234 \\ -1.6794775156491377 \\ 0.5082733976393556 \\ -0.25169818502348007 \end{pmatrix} \qquad x_{\frac{3m}{4}} = x_{18304} = \begin{pmatrix} -1.507841726609269 \\ -0.22060378567950611 \\ 3.343406739106548 \\ -1.9343835479751248 \\ 0.6029623016937389 \\ -0.332553486463615 \end{pmatrix}$$

$$x_{\frac{m}{2}} = x_{12203} = \begin{pmatrix} -1.4631521073720928 \\ -0.21528246944462 \\ 3.1935259611955886 \\ -1.850652841976869 \\ 0.5718681918959697 \\ -0.30601588147387093 \end{pmatrix} \qquad x_m = x_{24406} = \begin{pmatrix} -1.5297081304989302 \\ -0.22320747880544625 \\ 3.416742639117708 \\ -1.9753525554732039 \\ 0.6181764908218194 \\ -0.34553820068193253 \end{pmatrix}$$

Промежуточные значения функционала:

$$f\left(x_{\frac{m}{4}}\right) = f(x_{6101}) = -13.837969386830299$$

$$f\left(x_{\frac{m}{2}}\right) = f(x_{12203}) = -14.074776746789134$$

$$f\left(x_{\frac{3m}{2}}\right) = f(x_{18304}) = -14.131440448625039$$

$$f(x_m) = f(x_{24406}) = -14.145004232600975$$

Значение функционала в точке $x_{\text{точ}}$: $f(x_{\text{точ}}) = -14.149271102339297$ (точное решение)

Погрешности метода градиента:

$$\Delta(x_m, x_{\text{точ}}) = \begin{pmatrix} |x_{m1} - x_{\text{точ }1}| \\ \vdots \\ |x_{m6} - x_{\text{точ }6}| \end{pmatrix} = \begin{pmatrix} 0.020940295155866817 \\ 0.0024934188002600777 \\ 0.07022990151928266 \\ 0.03923384538125951 \\ 0.014569821928930526 \\ 0.012434772064819688 \end{pmatrix}$$

$$\Delta(f(x_m), f(x_{\text{точ}})) = 0.004266869738321688$$

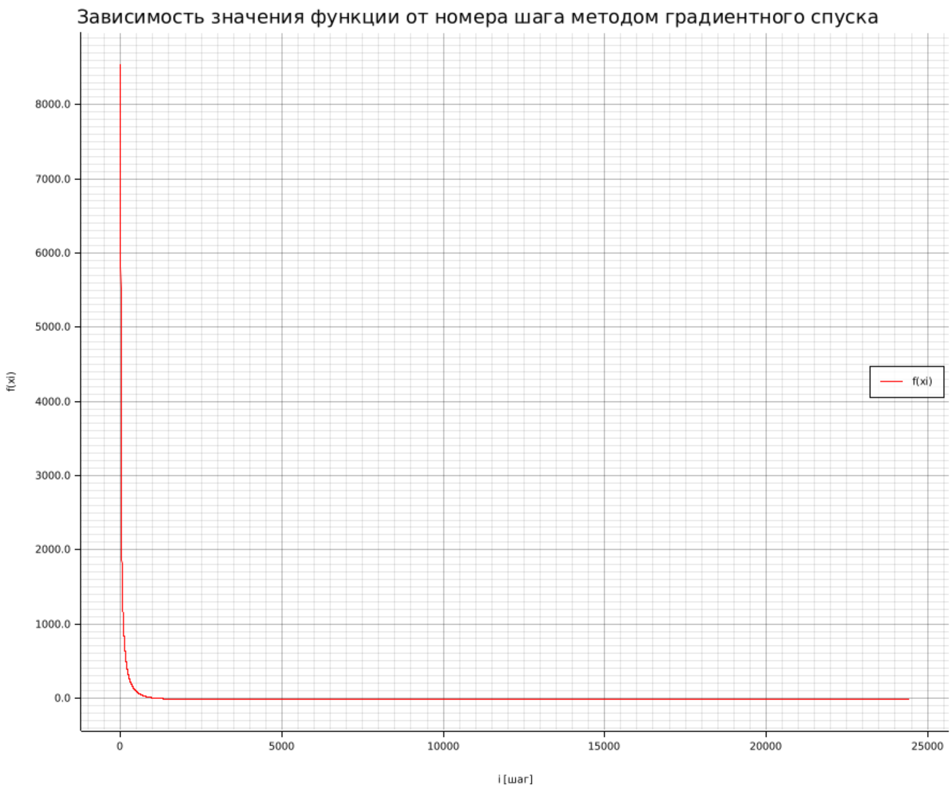

Рисунок 1: График зависимости значения функции от номера шага методом градиентного спуска

Следует отметить, что график не точно отражает реальное значение f(x) при приближении к точному значению.

Приложения

```
pub type FloatingType = f64;

pub mod lab1 {
    use nalgebra::{ArrayStorage, Const, DimMin, SquareMatrix, Vector};

    use crate::FloatingType;

    pub type __GenericSquareMatrix<const N: usize> =
        SquareMatrix<FloatingType, Const<N>, ArrayStorage<FloatingType, N, N>>;
    pub type __GenericVector<const N: usize> =
        Vector<FloatingType, Const<N>, ArrayStorage<FloatingType, N, 1>>;

    pub fn new_positive_definite_matrix<const N: usize>(
        min: FloatingType,
        max: FloatingType,
    ) -> __GenericSquareMatrix<N>
    where
        Const<N>: DimMin<Const<N>, Output = Const<N>>,
    {
        loop {
            let m = (max - min) * __GenericSquareMatrix::<N>::new_random()
                + __GenericSquareMatrix::<N>::from_element(min);
            if m.determinant() > 0 as FloatingType {
                return m.transpose() * m;
```

```rust
            }
        }
    }

    pub fn gradient_method<const N: usize, F>(
        x0: &__GenericVector<N>,
        lambda: FloatingType,
        epsilon: FloatingType,
        f_prime: F,
    ) -> Vec<__GenericVector<N>>
    where
        F: Fn(&__GenericVector<N>) -> __GenericVector<N>,
    {
        let mut x_log = vec![*x0];
        loop {
            let x = x_log.last().unwrap();
            let x_next = x - lambda * f_prime(x);
            if (x_next - x).norm() < epsilon {
                break;
            } else {
                x_log.push(x_next)
            }
        }
        x_log
    }
}
```