Тема 16 Методы минимизации с переменной метрикой

- *Исходные предпосылки*
- *Общий алгоритм методов с переменной метрикой*
- *Основные методы*

Недостатки и достоинства метода Ньютона

- Описанный в предыдущей лекции метод Ньютона на каждой итерации требует вычисления вектора градиента и матрицы вторых производных. Их приходится вычислять на основе разделенных разностей, а это требует значительных затрат.

$$\overline{x}^{k+1} = \overline{x}^{k} - z_m G^{-1}\left(\overline{x}^{k}\right)\nabla f\left(\overline{x}^{k}\right)$$

- Из-за этого трудоемкость каждой итерации метода Ньютона становится непомерно большой, хотя количество самих итераций в этом методе теоретически меньше, чем в других методах.
- Возникает вопрос – а возможно ли построить метод минимизации, который по количеству итераций не уступал бы методу Ньютона, но затраты на каждой итерации на вычисление нужного направления были бы значительно меньшими.

Основные предпосылки

- С другой стороны привлекательной является *идея адаптивного управления* спуском к минимуму. Как известно, система адаптивного управления имеет свойство накапливать информацию о предыдущем поведении управляемого объекта и, используя эту информацию, совершенствовать заложенный в ней алгоритм управления.
- В приложении к рассматриваемой нами задаче оптимизации *адаптивный алгоритм поиска* минимума должен как-то накапливать информацию, полученную в результате выполненных начальных спусков таким образом, чтобы постепенно приближаться к теоретически оптимальному выбору очередного направления обоснованного в методе второго порядка Ньютона.

Идея алгоритмов

- В рассмотренном ниже классе методов с переменной метрикой реализована идея адаптивного управления.
- Информация о предыдущих итерациях накапливается в специальной матрице *H*, которая по мере накопления информации постепенно превращается в обратную матрицу Гессе

$$H^k \xrightarrow[k\to\infty]{} G^{-1}(x^k)$$

- При этом матрица вторых производных на каждом шаге не пересчитывается.

Многообразие и перспективность алгоритмов

- Процесс обработки и накопления информации, получаемой на каждой новой итерации можно реализовать разными способами. Поэтому на сегодняшний день было предложено множество различных методов, которые мы рассмотрим ниже.
- Следует так же отметить, что на сегодняшний день предложен метод сходящийся за меньшее число итераций, чем метод Ньютона, хотя и использующий только матрицу вторых производных Гессе.
- [Хромова Л.Н. Об одном методе минимизации с кубической скоростью сходимости. – Вестник Моск. ун-та Сер. Вычислит. Матем. и кибернетика, 1980, №3].
- Поэтому возможно построение методов с переменной метрикой имеющих кубическую скорость сходимости, т. е. значительно более быстрых, чем метод Ньютона.

Общий алгоритм методов с переменной метрикой

- Методами с переменной метрикой называется широкий класс эффективных градиентных методов, в которых направление очередного спуска вычисляется по формуле

$$\vec{d}^{\,k} = -H_k \vec{g}^{\,k}$$

$$\vec{g}^{\,k} = \nabla f(\vec{x}^{\,k}) = \begin{bmatrix} g_1^k \\ \dots \\ g_n^k \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f(\vec{x}^{\,k})}{\partial x_1} \\ \dots \\ \dfrac{\partial f(\vec{x}^{\,k})}{\partial x_n} \end{bmatrix}$$

$$H_k = H_{k-1} + A_k$$

- *H - положительно определенная симметричная* матрица

Сравните с методом Ньютона

$$\vec{d}^{\,k} = -H_k \vec{g}^{\,k} \qquad \vec{d}^{\,k} = -G_k^{-1} \vec{g}^{\,k}$$

$$H_k = H_{k-1} + A_k$$

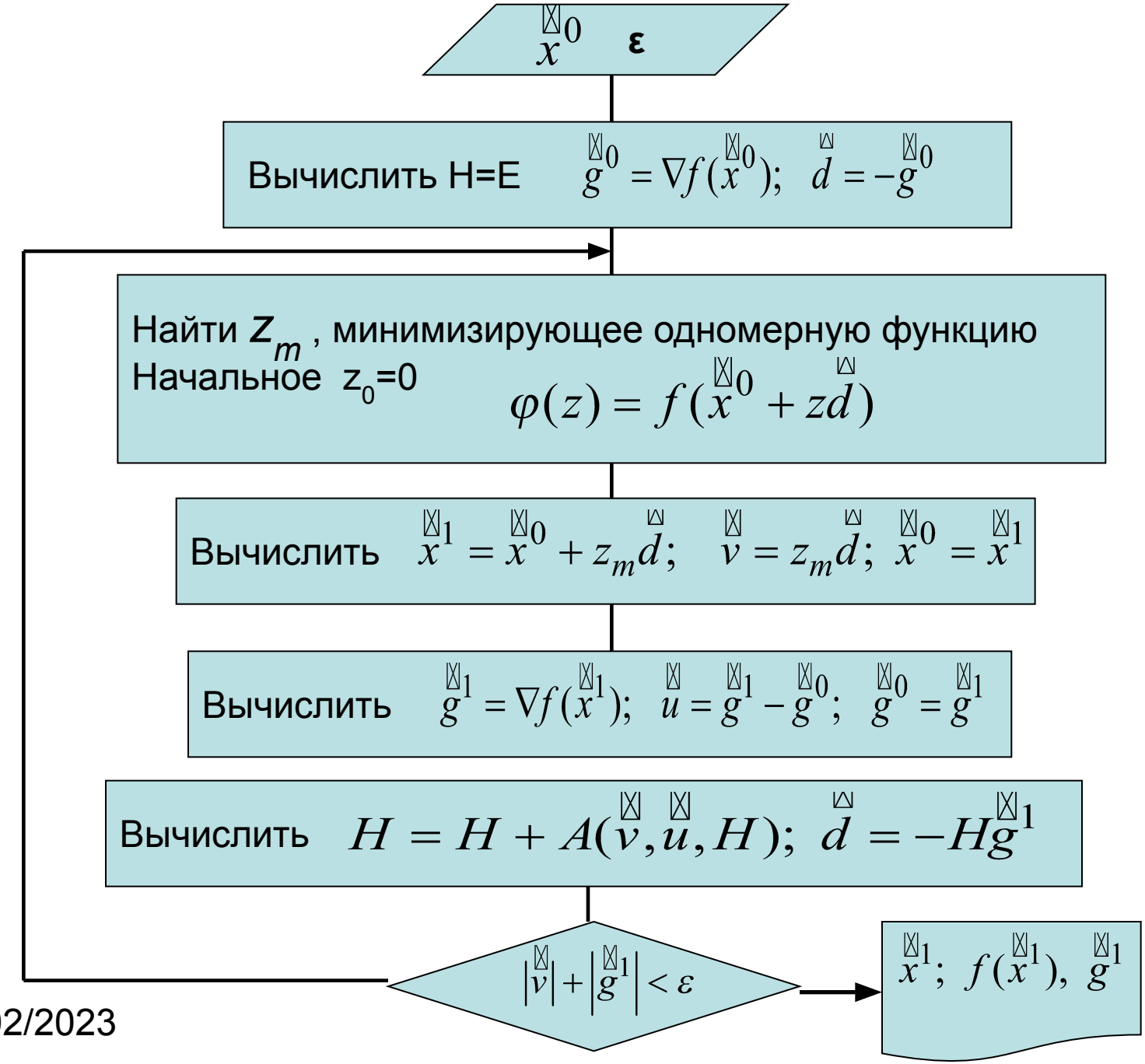
- A_k корректирующая матрица, рассчитываемая по результатам предыдущих спусков таким образом, чтобы обеспечить сходимость

$$H^k \xrightarrow[k\to\infty]{} G^{-1}(\vec{x}^{\,k})$$

Последовательность вычислений

- На 1-м шаге следует положить $H_0=E$ $\vec{g}^0 = \nabla f(\vec{x}^0)$
- Делаем очередной одномерный спуск в направлении $\vec{d}^k = -H_k \vec{g}^k$ $\vec{x}^{k+1} = \vec{x}^k + z_m \vec{d}^k$
- делается пересчет корректирующей матрицы $A_{k+1}(\vec{v},\vec{u},H)$ $\vec{v} = z_m \vec{d}^k,\ \vec{u} = \vec{g}^{k+1} - \vec{g}^k$
- Если $|z_m| + |\vec{g}^{k+1}| > \varepsilon$ то производится вычисление нового направления спуска $H_{k+1}=H_k+A_{k+1}$;

$$\vec{d}^{k+1} = -H_{k+1}\vec{g}^{k+1}$$


$\vec{x}^0$ ε
Вычислить H=E $\vec{g}^0 = \nabla f(\vec{x}^0);\ \vec{d} = -\vec{g}^0$
Найти z_m , минимизирующее одномерную функцию
Начальное z_0=0
$\varphi(z) = f(\vec{x}^0 + z\vec{d})$
Вычислить $\vec{x}^1 = \vec{x}^0 + z_m\vec{d};\ \vec{v} = z_m\vec{d};\ \vec{x}^0 = \vec{x}^1$
Вычислить $\vec{g}^1 = \nabla f(\vec{x}^1);\ \vec{u} = \vec{g}^1 - \vec{g}^0;\ \vec{g}^0 = \vec{g}^1$
Вычислить $H = H + A(\vec{v}, \vec{u}, H);\ \vec{d} = -H\vec{g}^1$
$|\vec{v}| + |\vec{g}^1| < \varepsilon$
$\vec{x}^1;\ f(\vec{x}^1),\ \vec{g}^1$

Общая характеристика методов

- В результате методы получаются *асимптотически второго порядка*.
- Матрица Гессе не пересчитывается на каждом спуске, и в результате *вычислительные затраты на итерацию аналогичны как и в простых градиентных методах*.
- Однако, в отличие от метода Ньютона, в котором минимум квадратичной функции находится за один спуск, методы с переменной метрикой *гарантируют нахождение минимума квадратичной функции за n спусков* (*n* – размерность вектора).

Рекомендация

- Обычно при реализации одномерного спуска используется метод кубической или квадратичной параболы. При этом, для уменьшения вычислительных затрат, рекомендуется делать только одну - две итерации, обеспечивая лишь получение улучшенного значения функции.
- Оказывается дополнительная работа, выполняемая для полного завершения одномерного спуска (с большой точностью) не оправдывает себя.
- Теоретически это означает, что метод как бы утрачивает свойство сходимости за *n* шагов для квадратичной функции, но практически остается эффективным и быстрым для произвольных функций

Метод ДАВИДОНА-ФЛЕТЧЕРА-ПАУЭЛЛА

- Один из первых и эффективных методов с переменной метрикой, на основе выше сформулированных Давидоном принципов предложен Флетчером и Пауэллом (1963).
- Матрица $A_к$ рассчитывается по формулам

$$A = \frac{\bar{v}\,\bar{v}^T}{\bar{v}^T\,\bar{u}} - \frac{H\,\bar{u}\,\bar{u}^T\,H}{\bar{u}^T\,H\,\bar{u}}$$

$$\bar{v} = z_m \bar{d}^k, \quad \bar{u} = \bar{g}^{k+1} - \bar{g}^k$$

Поясним вычисление матрицы A на примере двух переменных

$$\vec{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}; \quad \vec{v}^T = \begin{bmatrix} v_1 & v_2 \end{bmatrix}; \quad \vec{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}; \quad \vec{u}^T = \begin{bmatrix} u_1 & u_2 \end{bmatrix};$$

$$\vec{v}\vec{v}^T = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \begin{bmatrix} v_1 & v_2 \end{bmatrix} = \begin{bmatrix} v_1 v_1 & v_1 v_2 \\ v_2 v_1 & v_2 v_2 \end{bmatrix}$$

Получается матрица

$$\vec{v}^T u = \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = v_1 u_1 + v_2 u_2$$

Получается число

Вычисление матрицы A на примере двух переменных

$$H = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}$$

$$H\vec{u} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} h_{11}u_1 + h_{12}u_2 \\ h_{21}u_1 + h_{22}u_2 \end{bmatrix}$$

$$\vec{u}^T H = \begin{bmatrix} u_1 & u_2 \end{bmatrix}\begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} = \begin{bmatrix} h_{11}u_1 + h_{12}u_2 & h_{21}u_1 + h_{22}u_2 \end{bmatrix}$$

$$H\vec{u}\vec{u}^T H = \begin{bmatrix} h_{11}u_1 + h_{12}u_2 \\ h_{21}u_1 + h_{22}u_2 \end{bmatrix}\begin{bmatrix} h_{11}u_1 + h_{12}u_2 & h_{21}u_1 + h_{22}u_2 \end{bmatrix}$$

$$\vec{u}^T H\vec{u} = \begin{bmatrix} h_{11}u_1 + h_{12}u_2 & h_{21}u_1 + h_{22}u_2 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Программная реализация вычисления матриц H+A

```
Procedure PREOBR;
var
hu,uh,u,v: array[1..n] of real;
vu,uhu,uh:real;
begin
vu:=0;for i:=1 to n do begin
        v[i]:=zm*d[i];
        u[i]:=g1[i]-g0[i]; //
        vu:=vu+v[i]*u[i]
    end;
uhu:=0; for i:=1 to n do begin
        uh:=0;
        for j:=1 t n do
        uh:=uh+u[j]*H[j,i]; //
uhu:=uhu+uh*u[i]
     end;
```

$$\vec{v}^T \vec{u}$$

$$\vec{u}^T H \vec{u}$$

Программная реализация (продолжение)

```
for i:=1 to n do begin
    hu[i]:=0; uh[i]:=o;
  for k:=1 to n do begin
hu[i]:=hu[i]+H[i,k]*u[k];
    uh[i]:=uh[i]+u[k]*H[k,i];
              end
        end;
for i:=1 to n do
for j:=1 to n do begin
 Aij:=v[i]*v[j]/vu-hu[i]*uh[j]/uhu;
 H[i,j]:=H[i,j]+Aij;
           end
end;
```

$H \vec{u}$

$\vec{u}^T H$

Общий вид матрицы А

- В 1979г. Гринсдадт Дж. предложил общее выражение, задающее в зависимости от выбираемой произвольной симметричной матрицы M семейство методов с переменной метрикой

$$A = \frac{1}{u^T H u}\left\{ v u^T M + M u v^T - H u u^T M - M u u^T H - \frac{\left[u^T v - u^T H u\right]}{u^T H u} H u u^T H \right\}$$

- В последствии исходя из него были предложены различные методы с переменной метрикой, наиболее эффективным зарекомендовал себя метод Гольдфарба

Метод ГОЛЬДФАРБА (1970)

$$A = \frac{-v\,u^T H + H\,u\,v^T + \left(1 + \frac{u^T H\,u}{u^T v}\right) v\,v^T}{u^T v}$$

Он более устойчив к ошибкам вычислений, чем ДФП

Метод ПРОЕКТИВНОГО ГРАДИЕНТА

$$A = -\frac{(H\vec{u})(H\vec{u})^T}{\vec{u}^T H \vec{u}}$$

После N шагов матрица должна восстанавливаться: H=E.

Метод менее громоздок

Метод МАК-КОРМИКА 1

$$A = \frac{\left(\vec{v} - H\vec{u}\right)\vec{v}^{T}}{\vec{v}^{T}\vec{u}}$$

Метод МАК-КОРМИКА 2

$$A = \frac{(\hat{v} - H\hat{u})(\hat{u}H)}{u^T H u}$$

Метод ГРИНСТАДТА

$$A = \frac{\vec{v}\,\vec{u}^T H + H\,\vec{u}\,\vec{v}^T - \left(1 + \frac{\vec{u}^T\,\vec{v}}{\vec{u}^T H\,\vec{u}}\right) H\,\vec{u}\,\vec{u}^T H}{\vec{u}^T H\,\vec{u}}$$

Сравнение методов по эффективности

- Среди методов нулевого порядка наибольшей эффективностью обладают *метод Розенброка* и *метод Нелдера Мида*
- Если же функция гладкая, т.е. имеет непрерывные производные, то наиболее эффективным себя зарекомендовали *метод Гольдфарба и метод ДФП* (они в 2-6 раз эффективнее по быстродействию чем метод Розенброка).
- На самом деле многое зависит от вида конкретной оптимизируемой функции. Особенно если количество оптимизируемых переменных больше 4-5 нужно иметь в запасе несколько разнообразных методов, а процесс оптимизации строить на основе интерактивных программ.

Конец