In this section we will look at nonlinear optimization

- Single variable NLP
  - Review
  - Golden Section Search

- Multi variable NLP
  - Steepest Ascent Method
  - Euler-Lagrange Multipliers
  - Kuhn-Tucker Conditions

- Quadratic Programming
  - Wolfe’s extended Simplex Method
Classes of Optimization Problems

Optimization

unconstrained
- ...
- real-valued
- integer

constrained
- linear
- non-linear
- quadratic
- ...
- ...
- ...
In general, non-linear problems are non-convex. However, there are important convex sub-classes of NLP.
Non-Linear Programming

\[
\begin{align*}
\text{max/min } & \ f(\bar{x}) \ \text{subject to } \ C(\bar{x}) \\
\bar{x} & \\
\text{where } & \ f(\bar{x}) = f(x_1, \ldots x_n) \\
\text{is a non-linear function } & \ f : R^n \rightarrow R \text{ and} \\
C(\bar{x}) & = c_1(x_1, \ldots x_n) \land \ldots \land c_k(x_1, \ldots, x_n) \\
\text{is a conjunction of linear or non-linear constraints.}
\end{align*}
\]
Practical Example of a NLP

A bloodbank wants to position a central dispatch facility.

There are two hospitals a,b located at c(a)=(3,9) and c(b)=(6,11) which must be reachable quickly and should therefore not be at a distance of more than 5 units (5 km) from the dispatch.

The city has three donor stations m, n, o which operate for the following average number of days per month: e(m)=3, e(n)=5, e(o)=12. These stations are located at the following positions in the x-y plane: c(m)=(12, 3); c(n)=(14, 7); c(o)=(21, 0).

We want to minimize the average distance travelled between dispatch and donor stations (assuming one tour per open day and station).

\[
\min_{p=(x,y)} 3 \cdot d(p, c(m)) + 5 \cdot d(p, c(n)) + 12 \cdot d(p, c(c))
\]

subject to \( d(p, c(a)) \leq 5 \land d(p, c(b)) \leq 5 \)

where \( d((x_1, y_1), (x_2, y_2)) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \)
Consider the following (supposed) objective functions:

- local = global
- local $\neq$ global (most non-linear functions)

Non-linear constraints: feasible region can have any shape in particular non-convex set.

- gradient descent can be used
- gradient descent gets trapped in local mimima
Convex and Concave Functions

\( f(\bar{x}) \) is a **convex** function on a convex set \( S \) if
\[
\forall x', x'' \in S, \quad 0 \leq c \leq 1 : \quad f(cx' + (1 - c)x'') \leq cf(x') + (1 - c)f(x'')
\]

\( f(\bar{x}) \) is a **concave** function on a convex set \( S \) if
\[
\forall x', x'' \in S, \quad 0 \leq c \leq 1 : \quad f(cx' + (1 - c)x'') \geq cf(x') + (1 - c)f(x'')
\]

Linear functions are simultaneously concave and convex!
Let $f(x^*)$ be the objective of a maximization NLP with a convex feasible region $S$.

If $f(x^*)$ is concave on $S$, then any local optimum is a global optimum.

Let $f(x^*)$ be the objective of a minimization NLP with a convex feasible region $S$.

If $f(x^*)$ is convex on $S$, then any local optimum is a global optimum.
(Unconstrained) Single Variable NLP

$$\max / \min \quad f(x)$$

$$x \in [a, b]$$

- $f'(x) = 0$: maximum
- $f'(x) = 0$: saddle
- Non-existent

- $f'(x) = 0$: minimum
if $f'(x)=0$ then

- if the first non-vanishing derivative at $x$ is of **even** order and
  - positive then $x$ is a *local* minimum
  - negative then $x$ is a *local* maximum
- if the first non-vanishing derivative at $x$ is of **odd** order
  then $x$ is **not** an extremum

$$f(x) = x^3$$
$$f'(x) = 3x^2 \quad f'(0) = 0$$
$$f''(x) = 6x \quad f''(0) = 0$$
$$f'''(x) = 6 \quad f'''(0) = 6 \quad \text{odd!}$$
Endpoints of Interval

Of course, the optimum could be found at one of the endpoints of the given interval (domain) for the variable.

It is easy to see that
- $x_0$ is a *local* minimum if $f'(x_0) > 0$
- $x_0$ is a *local* maximum if $f'(x_0) < 0$
- $x_1$ is a *local* maximum if $f'(x_1) > 0$
- $x_1$ is a *local* minimum if $f'(x_1) < 0$
Unimodal Functions

Systematic numeric search can be used as an alternative. If we know that we have only one extremum in the search range, repeated splitting of the interval can narrow it until the extremum is found.

A function $f$ is unimodal on $[x_0, x_1]$, if there is a point $x$, $x_0 < x < x_1$ such that $f$ is strictly increasing on $[x_0, x]$ and strictly decreasing on $[x, x_1]$. 
A unimodal function $f(x)$ on $[x_0, x_1]$ can be maximized by search:

(1) evaluate $f(x)$ at two sample points $a$, $b$ in $l=[x_0, x_1]$
(2a) if $f(a)<f(b)$ set $l := [a,x_1]$
(2b) if $f(a)\geq f(b)$ set $l := [x_0,b]$
(3) if $d=(a-b)$ is sufficiently small then return
else repeat from (1)

The search interval is called the “interval of uncertainty”
Golden Section Search

tries to make the search as effective as possible by reducing the number of function points to be sampled.

In each iteration only one new point needs to be evaluated!

For an interval $I=[x_0, x_1]$ sample $f$ at

$$a = x_1 - r(x_1 - x_0)$$
$$b = x_0 + r(x_1 - x_0)$$

solve $r^2 + r = 1$

$$r = \frac{\sqrt{5} - 1}{2} \approx 0.618$$
Golden Section Properties

Let the initial interval be $I = [x_0, x_1]$

$f$ is sampled at $a = x_1 - r^*(x_1 - x_0)$ and $b = x_0 + r^*(x_1 - x_0)$

Assume $f(a) < f(b)$. Then $I' = [a, x_1]$ with length $x_1 - a = r^*(x_1 - x_0)$.

The new sample points are then

$a' = x_1 - r^*(x_1 - a) = x_1 - r^*r^*(x_1 - x_0)$  // with $x_1 - a = r^*(x_1 - x_0)$.

$b' = a + r^*(x_1 - a)$

Note that $a' = b$, because of $r^*r = (1-r)$ (by definition)

$a' = x_1 - (1-r)(x_1 - x_0) = x_1 - (x_1 - x_0) + r(x_1 - x_0) = x_0 + r(x_1 - x_0) = b$

The case $f(a) \geq f(b)$ is analyzed analogously.

Note: The length of the uncertainty interval after $k$ iterations is $r^k*(x_1 - x_0)$
NLP with Several Variables

For a local extremum of a function of several variables all partial derivatives need to be zero:

\[
\text{if } \bar{x} \text{ is a local extremum for } f(\bar{x}) \text{ then } \\
\forall i: \frac{\partial f(\bar{x})}{\partial x_i} = 0
\]

Any such point is called a “stationary point”.

Remember:
The partial derivative is defined as

\[
\frac{\partial f}{\partial x_i} = \lim_{\epsilon \to 0} \frac{f(x_1, \ldots, x_i + \epsilon, x_{i+1}, \ldots, x_n) - f(x_1, \ldots, x_n)}{\epsilon}
\]

To find it, consider \(x[1]...x[i-1],x[i+1],...,x[n]\) as constants

\[
\frac{\partial x^2 y^3}{\partial x} = 2xy^3 \quad \frac{\partial x^2 y^3}{\partial y} = 3x^2y^2
\]
To define sufficient conditions for a local extremum, we use the Hessian Matrix.

The Hessian Matrix is the square matrix of all partial 2nd order derivatives of a function.

Let \( f(\vec{x}) = f(x_1, \ldots, x_n) \).

The Hessian of \( f \) is the \( n \times n \) Matrix in which the entry at position \((i, j)\) is

\[
\frac{\partial^2 f}{\partial x_i \partial x_j}
\]

To find the second order derivative first differentiate for \( x[i] \), then differentiate the result for \( x[j] \).
Determinants

The determinant of a $1 \times 1$ Matrix $A = \begin{bmatrix} a_{11} \end{bmatrix}$ is
\[ \det A = a_{11} \]

The determinant of a $2 \times 2$ Matrix $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ is
\[ \det A = a_{11}a_{22} - a_{12}a_{21} \]

To compute the determinant for any larger matrix use the following development: pick an arbitrary row number $i \in 1 \ldots n$ and compute
\[ \det A = (-1)^{i+1} a_{i1} (\det A_{\text{ij}}) + \ldots + (-1)^{i+m} a_{im} (\det A_{\text{jm}}) \]

This is called "cofactor expansion".

$A_{ij}$ denotes the sub matrix of $A$ obtained by deleting row $i$ and column $j$ from $A$. 
Leading Principal Minors

The $k$-th leading principal minor of the $n \times n$ Matrix $A$ is obtained by computing the determinant of the matrix obtained by deleting the last $(n - k)$ rows and columns of $A$.

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

$$3\text{rd} \quad \Rightarrow \quad \begin{bmatrix} 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

The 2-nd leading principal minor of $A$ is

$1 \times 6 - 2 \times 5 = -4$
Let $H[k](x)$ denote the $k$-th leading principal minor of the Hessian matrix of $f(x_1, ..., x_n)$ evaluated at point $x$.

- A stationary point $x_0$ is a local minimum if and only if $H[k](x_0) > 0$ for $k=1,2,...,n$

- A stationary point $x_0$ is a local maximum if and only if $H[k](x_0) \neq 0$ for $k=1,2,...,n$ and $\text{sign}(H[k](x_0)) = (-1)^k$ for $k=1,2,...,n$
Suppose we want to search for an extremum (stationary point) of an NLP with several variables. In which direction shall we let the search proceed?
Gradient Vectors

The simplest idea is to always proceed in the “most promising” direction.

This means to proceed in the direction of the steepest slope of the objective function. (positive slope for maximization, negative for minimization).

This direction is given by the gradient vector

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}$$

Note that the gradient vector at $x_0$ is always perpendicular to the level curve at $x_0$!

$$f(x, y) = x^2 + y^2$$

$$\nabla f(x, y) = [2x, 2y]$$

$$\nabla f(2, 2) = [4, 4]$$

Problem: This direction is only valid for the immediate neighbourhood of $x_0$ and therefore only for small steps.
Steepest Ascent Method

To maximize \( f(\bar{x}) = f(x, \ldots x_n) \)

1. choose an arbitrary point \( x_0 \)
2. compute the gradient vector \( \nabla f(x_0) \)
3. solve the best step width
   \[
   \max_t f(x_0 + t \cdot \nabla f(x_0)) \quad \text{subject to } t \geq 0
   \]
4. if \( \nabla f(x_0) \) is small enough, terminate
   else repeat from step (1) with \( x_0 := x_0 + t \cdot \nabla f(x_0) \)

**Question:** How can the max NLP in step (3) be solved ?!?!
Steepest Ascent Method

To maximize \( f(\bar{x}) = f(x_1, \ldots x_n) \)

1. choose an arbitrary point \( x_0 \)
2. compute the gradient vector \( \nabla f(x_0) \)
3. solve the best step width
   \[
   \max_{t} f(x_0 + t \cdot \nabla f(x_0)) \quad \text{subject to } t \geq 0
   \]
4. if \( \nabla f(x_0) \) is small enough, terminate
   else repeat from step (1)

The maximization in step (3) is a unconstrained NLP of one variable \( t \). If it cannot be solved analytically, the search methods described above can be used.
Constraint NLP

So far, all our NLPs have been unconstrained (or on an interval). For complex constraints the previous ideas are of very restricted use.

Consider the constraint:
\[ \sin(2x) + \cos(3y) = 1 \]

This defines a non-convex feasible space.
It is unclear how we can search in non-convex space!
A constraint NLP has the form

\[
\begin{align*}
\text{max/min} \quad z &= f(\bar{x}) = f(x_1, \ldots, x_n) \\
g_1(\bar{x}) &= b_1 \\
\vdots &= \vdots \\
g_n(\bar{x}) &= b_n \\
\end{align*}
\]

subject to

where, \( f(x^*) \) and \( g_i(x^*) \) are non-linear functions.
Lagrange Multipliers

A constraint NLP can be solved by converting it into an equivalent unconstrained NLP:

\[
\begin{align*}
\text{max/min} \quad & z = f(\bar{x}) = f(x_1, \ldots, x_n) \\
& g_1(\bar{x}) = b_1 \\
\text{subject to} \quad & \vdots = \vdots \\
& g_n(\bar{x}) = b_n
\end{align*}
\]

Associate multipliers \( \lambda \) with the constraint and solve instead:

\[
\begin{align*}
\text{max/min} \quad & f(\bar{x}, \bar{\lambda}) = f(x_1, \ldots, x_n, \lambda_1, \ldots, \lambda_n) = \\
& f(\bar{x}) + \lambda_1 (b_1 - g_1(\bar{x})) + \ldots + \lambda_n (b_n - g_n(\bar{x}))
\end{align*}
\]

This increases the dimension of the problem, but turns it into an unconstrained problem!
Multiplier Interpretation

\[
\begin{align*}
\text{max/min} \quad f(\bar{x}, \bar{\lambda}) &= f(x_1, \ldots, x_n, \lambda_1, \ldots, \lambda_n) = \\
&= f(\bar{x}) + \lambda_1 (b_1 - g_1(\bar{x})) + \ldots + \lambda_n (b_n - g_n(\bar{x}))
\end{align*}
\]

- At an extremum \((\bar{x}_e, \bar{\lambda}_e)\) of \(f(\bar{x}, \bar{\lambda})\) we must have \(\frac{\partial f(\bar{x}, \bar{\lambda})}{\partial \lambda_i} = 0\) for all \(i\) because all partial derivatives must be zero.

- Therefore the extremum is a feasible point because it fulfills for all \(i\)

\[
\frac{\partial f(\bar{x}, \bar{\lambda})}{\partial \lambda_i} = b_i - g_i(\bar{x}) = 0
\]

- It also follows that since for all \(i\) : \(b_i - g_i(\bar{x}_e) = 0\)

at the extremum we have \(f(\bar{x}, \bar{\lambda}) = f(\bar{x})\) for any \(\lambda\).

- Therefore \(f(\bar{x}_e)\) is a solution of the original constraint problem.
Finding the Extremum

\[
\text{max/min } f(\bar{x}, \bar{\lambda}) = f(x_1, \ldots, x_n, \lambda_1, \ldots, \lambda_n) = \\
f(\bar{x}) + \lambda_1 (b_1 - g_1(\bar{x})) + \ldots + \lambda_n (b_n - g_n(\bar{x}))
\]

We know that at the extremum of \(f(\bar{x}, \bar{\lambda})\) we have

\[
\frac{\partial f(\bar{x}, \bar{\lambda})}{\partial x_1} = \ldots = \frac{\partial f(\bar{x}, \bar{\lambda})}{\partial x_k} = \frac{\partial f(\bar{x}, \bar{\lambda})}{\partial \lambda_1} = \ldots = \frac{\partial f(\bar{x}, \bar{\lambda})}{\partial \lambda_n} = 0
\]

This tells us how to find the extremum of the Lagrangian, but it is only a necessary, not a sufficient condition.

Remember saddle points!!!
Example

Find the closest point to the point (2,2) on a circle with radius 1 around (0,0)

minimize the Lagrangian \((x - 2)^2 + (y - 2)^2 + \lambda (x^2 + y^2 - 1)\)

(1) \(\frac{\partial f}{\partial x} = 2(x - 2) + 2\lambda x = 0\)

(2) \(\frac{\partial f}{\partial y} = 2(y - 2) + 2\lambda y = 0\)

(3) \(\frac{\partial f}{\partial \lambda} = x^2 + y^2 - 1 = 0\)

solve (1) and (2) for x and y \(x = y = \frac{2}{\lambda + 1}\)

and substitute into (3) \(\left(\frac{2}{1+\lambda}\right)^2 + \left(\frac{2}{1+\lambda}\right)^2 = 1\)

this yields \(\lambda = -1 \pm 2\sqrt{2}\) (for min/max) so for minimum \(x_e = y_e = \frac{1}{\sqrt{2}}\)
It is evident that the solution to the constraint NLP must be at the point where the level curve and the constraint curve touch tangentially. At any other point we could “slide along” the constraint satisfaction curve to find a better objective value.

(In the illustration better values are at the interior)
Geometric Interpretation

At the extremum the gradient vectors of $f$ and $g$ are parallel, for multiple constraints the gradient of $f$ is a linear combination of the gradients of the constraint curves $g_i$.

$$\nabla f(\bar{x}) = \sum \lambda_i \nabla g_i$$

We can therefore interpret the multiplier as a “shadow price”.

Sufficient Conditions

However, for certain simpler cases we can give sufficient conditions:

Let all constraints $g_i$ be linear functions.

- **Maximization**
  
  If $f(x)$ is a concave function then any extremum of the Lagrangian is a maximum of $f(x)$ subject to $g_i$

- **Minimization**
  
  If $f(x)$ is a convex function then any extremum of the Lagrangian is a minimum of $f(x)$ subject to $g_i$
Kuhn-Tucker Conditions

\[
\begin{align*}
\text{max/min} & \quad f(\bar{x}) = f(x_1, \ldots, x_n) \\
g_1(\bar{x}) & \leq b_1 \\
\text{subject to} & \quad \vdots \\
g_n(\bar{x}) & \leq b_n
\end{align*}
\]

For NLPs of this form where \( f(x) \) is concave (max) or convex (min) we can find necessary and sufficient conditions Subject to regularity conditions on the constraints \( g_i(x) \) That are always fulfilled for linear constraints.
Kuhn-Tucker Necessary Conditions

The main K-T conditions can be viewed as a generalization of the conditions derived for Lagrange multipliers:

Maximization: at the optimum point $\bar{x}_e$ we must have
\[
\forall j : \frac{\partial f(\bar{x})}{\partial x_j} - \sum \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} = 0
\]
\[
\forall i : \lambda_i (b_i - g_i(\bar{x})) = 0
\]
\[
\forall i : \lambda_i \geq 0
\]

Minimization: at the optimum point $\bar{x}_e$ we must have
\[
\forall j : \frac{\partial f(\bar{x})}{\partial x_j} + \sum \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} = 0
\]
\[
\forall i : \lambda_i (b_i - g_i(\bar{x})) = 0
\]
\[
\forall i : \lambda_i \geq 0
\]
Kuhn-Tucker Main Condition

At \( \bar{x} \) we have the objective value \( f(\bar{x}) \). Now assume we increase \( x_j \) by \( \Delta \).

This will change the objective to \( f(\bar{x}) + \Delta \cdot \frac{\partial f(\bar{x})}{\partial x_j} \).

Consider the \( i \)-th constraint \( g_i(\bar{x}) \leq b_i \). Its left-hand side increases by \( \Delta \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} \).

This corresponds to decreasing the right-hand side: \( g_i(\bar{x}) \leq b_i - \Delta \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} \).

\( g_i \) has shadow price \( \lambda_i \): Total objective change is \( \Delta \cdot \left[ \frac{\partial f(\bar{x})}{\partial x_j} - \sum \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} \right] \).

\( \bar{x} \) is extremum: no change may happen \( \forall j : \frac{\partial f(\bar{x})}{\partial x_j} - \sum \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} = 0 \).
Necessary K-T Conditions

It is obvious that
\[ \lambda_i \neq 0 \] only if \( g_i \) is tight, i.e. \( g_i(\bar{x}) = b_i \)
\[ \lambda_i = 0 \] otherwise.

Why must we have \( \lambda_i \geq 0 \)?
\( \lambda_i \) is a generalized shadow prize...

(1) Because relaxing constraint \( i \) by \( \varepsilon \) from
\[ g_i(x) < b_i \] to \( g_i(x) < b_i + \varepsilon \) only adds points to the feasible region
the optimal objective value can only increase (or stay constant).

Assume \( \lambda_i \neq 0 \), i.e. constraint \( i \) is binding.

(2) Because \( \lambda_i \) is a shadow prize, relaxing \( g_i \) by (a positive) \( \varepsilon \) would
increase the objective value by approximately \( \varepsilon \cdot \lambda_i \).

\( (1 \& 2) \varepsilon \cdot \lambda_i \geq 0 \) for \( \varepsilon > 0 \), therefore \( \lambda_i \geq 0 \).
The geometric interpretation of K-T conditions is also a generalization of the Lagrange interpretation. The given conditions hold at a point if and only if the gradient of the objective function at $x$ is a linear combination of the weighted gradients of the constraints. The weight for the $i$-th constraint is 0 if the constraint is non-binding at $x$.

$$\nabla f(\bar{x}) = \sum \lambda_i \nabla g_i(\bar{x})$$
K-T Non-negative Conditions

Often it is required that the problem variables \( x_i \) are non-negative.

Maximization: \( \forall j: \frac{\partial f(\bar{x})}{\partial x_j} - \sum_i \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} + \mu_i = 0 \)

\( \forall i: \lambda_i (b_i - g_i(\bar{x})) = 0 \)

\( \forall j: \left[ \frac{\partial f(\bar{x})}{\partial x_j} - \sum_i \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} \right] \cdot x_j = 0 \)

\( \forall i: \lambda_i \geq 0, \quad \mu_i \geq 0 \)

Minimization: \( \forall j: \frac{\partial f(\bar{x})}{\partial x_j} + \sum_i \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} - \mu_i = 0 \)

\( \forall i: \lambda_i (b_i - g_i(\bar{x})) = 0 \)

\( \forall j: \left[ \frac{\partial f(\bar{x})}{\partial x_j} + \sum_i \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} \right] \cdot x_j = 0 \)

\( \forall i: \lambda_i \geq 0, \quad \mu_i \geq 0 \)

We will later refer to the first condition as the “main” condition. Note that the main condition could also be written as an inequality since \( \mu_j \geq 0 \). For minimization:

\( \forall j: \frac{\partial f(\bar{x})}{\partial x_j} + \sum_i \lambda_i \cdot \frac{\partial g_i(\bar{x})}{\partial x_j} \geq 0 \)
K-T Sufficient Conditions

The basic K-T conditions are only necessary.

Under some additional assumptions they are even sufficient.

• Maximization

  If $f$ is a concave function and all $g_i$ are convex functions.

• Minimization

  If $f$ is a convex function and all $g_i$ are convex functions.

This holds for the standard K-T conditions as well as for the non-negative K-T conditions.
Computational Considerations

So far we can solve NLPs only analytically or on the basis of search.

Lagrange multipliers and Kuhn-Tucker conditions

- do not change this situation.
- enabled us to solve constraint NLP by reduction to unconstrained NLP.

An interesting observation is that multipliers methods can (in an analytic) approach not only be used for functions, but also for functionals, i.e. they even be used to find an optimum function for a problem.

For an in-depth discussion of this point see Variational Methods in Optimization, D.R. Smith, Dover 1974.