Projekt Optymalizacja nieliniowa

Cz 2 Optymalizacja wielowymiarowa

Krawiec Piotr
Inżynieria i analiza danych, 3 Rok

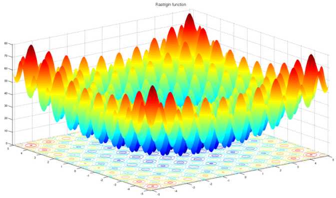
01/12/2021

Wstęp

Celem tej pracy jest omówienie metody gradientów sprzężonych Polaka-Ribiere'a, jej implementacja w R oraz rozwiązanie z jej pomoca zadania optymalizacyjnego.

Problem

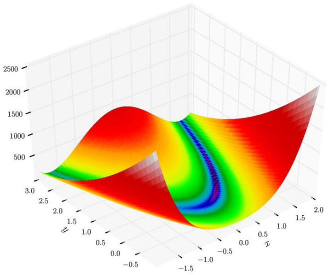
Istnieje wiele algorytmów optymalizacji co zrodziło potrzebę porównania ich, wyłonienia, który jest najlepszy. Do tego celu powstały funkcje testujące algorytmy optymalizacji [1] . Ich pierwszy zbiór został stworzony jako pakiet w programie Matlab przez Rody Oldenhuis i zawierał 50 funkcji testowych.

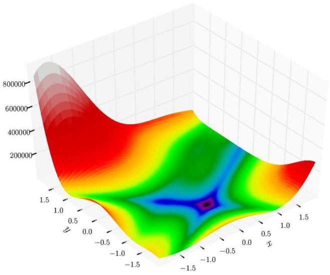


Rysunek 1: Funkcja Rastrigina

[1] https://en.wikipedia.org/wiki/Test_functions_for_optimization

Inne przykłady: Funkcja Rosenbrock'a oraz Goldstein'a-Price'a





Zadanie

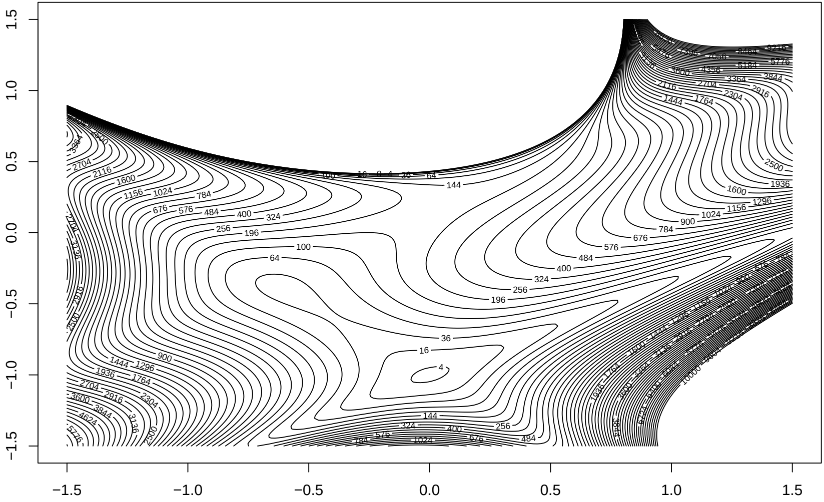
Zadaniem będzie znalezienie minimum funkcji Goldstein'a-Price'a o następującym równaniu:

$$f(x,y) = \left[1 + (x+y+1)^2\left(19 - 14x + 3x^2 - 14y + 6xy + 3y^2\right)\right]$$

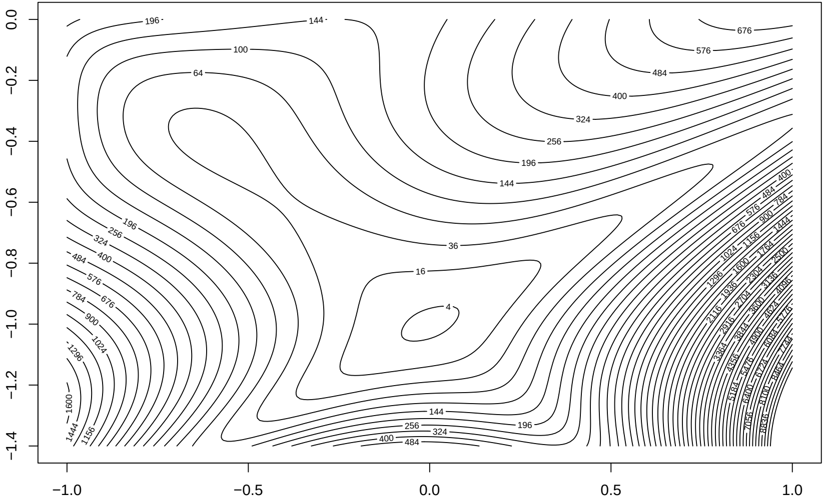
$$\left[30 + (2x-3y)^2(18 - 32x + 12x^2 + 48y - 36xy + 27y^2)\right]$$

```r
f <- function(x, y) {
  (1 + (x + y + 1)^2 *
  (19 - 14 + 3*x^2 - 14*y + 6*x*y + 3*y^2)) *
  (30 + (2*x - 3*y)^2*(18 - 32*x + 12*x^2
      + 48*y-36*x*y + 27*y^2))
}
```

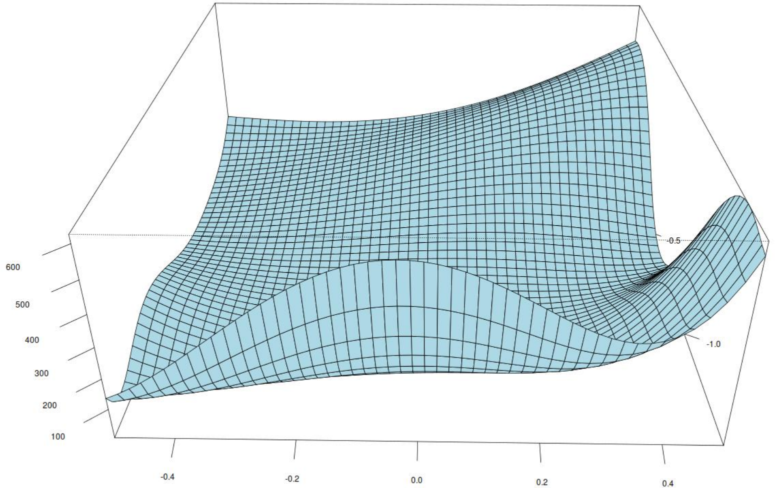
Wykres funkcji



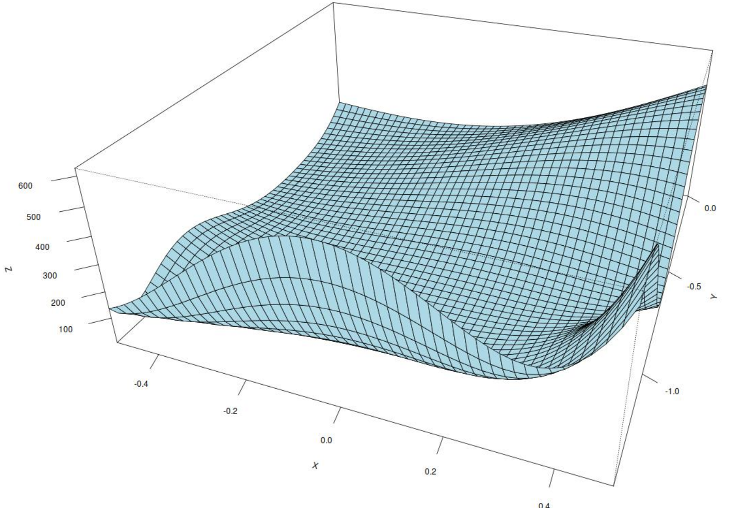
Wykres funkcji - zbliżenie na miejsce zerowe



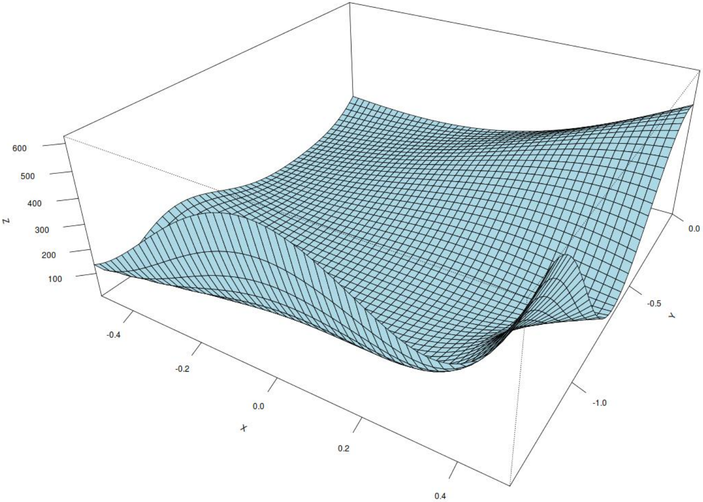
Wykres - 3D



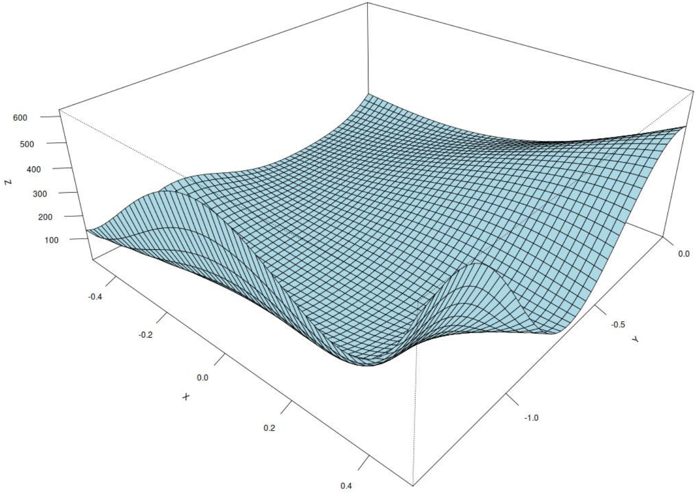
Wykres - 3D



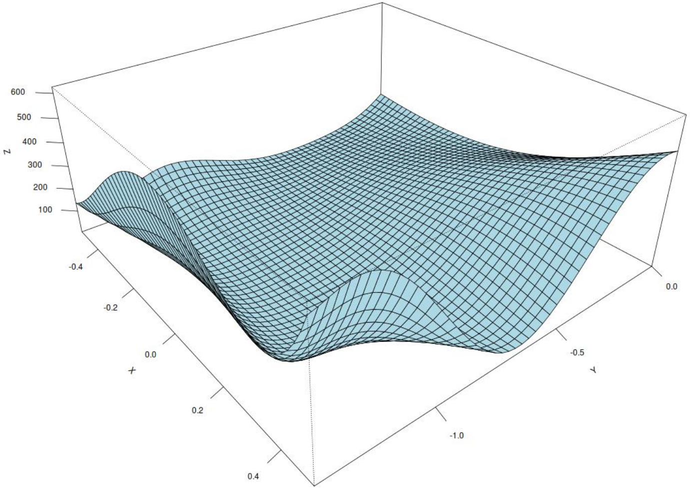
Wykres - 3D



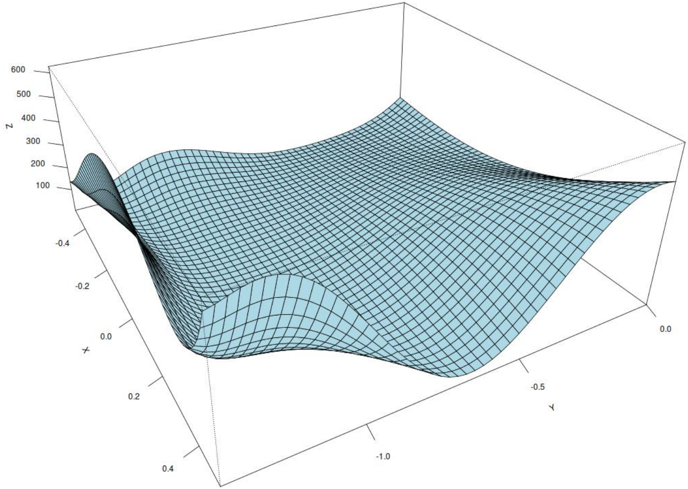
Wykres - 3D



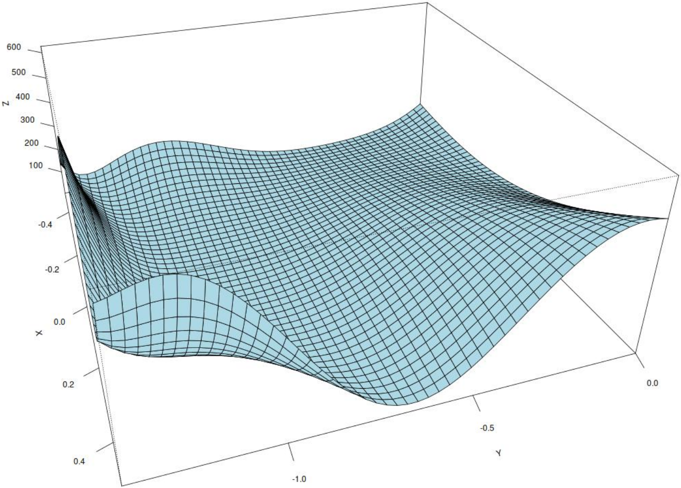
Wykres - 3D



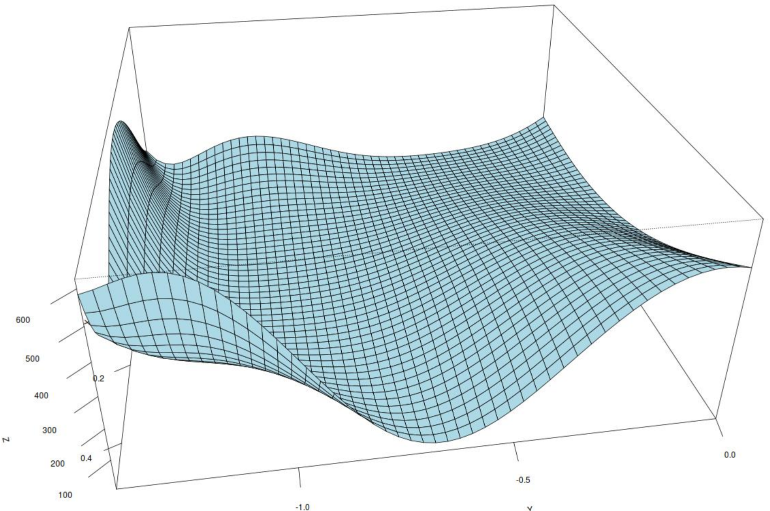
Wykres - 3D



Wykres - 3D



Wykres - 3D



Metoda gradientów sprzężonych Polaka-Ribiere'a

Jest to iteracyjna metoda optymalizacji bez ograniczeń. Odróżnia się od innych metod gradientowych tym, iż do oblizenia kolejnego kierunku poszukiwań korzysta z gradientu z poprzedniej iteracji (stąd sprzężenie w nazwie). Ogólna zasada jest podobna do metod gradientowych, w kroku pierwszym poszukujemy gradientu, który wskaże nam kierunek poszukiwania najmniejszego punktu:

$$d(0) = -\nabla f(x^{(0)})$$

Następnie w każdym z kolejnych $d^{(i+1)}$, uwzględniać będziemy β, które zawiera gradient.

$$d^{(i+1)} = -\nabla f(x^{(i+1)}) + \beta^{(i+1)} d^{(i)}$$

Gdzie:

$$\beta^{i+1} = \frac{\nabla f(x^{(i+1)})^T \left(\nabla f(x^{(i+1)}) - \nabla f(x^{(i)})\right)}{\nabla f(x^{(i)})^T \nabla f(x^{(i)})}$$

Metoda gradientów sprzężonych Polaka-Ribiere'a

I ostatecznie, kolejne przybliżenie szukanego minimum można znaleźć:

$$x^{(i+1)} = x^{(i)} + h^{(i)} * d^{(i)}$$

Przy czym, h - długość kroku można wyznaczyć na kilka sposobów. Najbardziej optymalnym jest aktualizacja tej wartości w każdej iteracji. Można to robić poszukując minium funkcji wzdłóż kierunku poszukiwań d.

$$g(\alpha) = f(x^{(i)} + \alpha d^{(i)})$$

Wtedy, dla dla znalezionego g_{min} i α_{min}:

$$h^{(i)} = \alpha_{min}$$

Minium funkcji g poszukuje się z pomocą metod optymalizacji jednowymiarowej.

Metody optymalizacji jednowymiarowej - mały (wielki) problem

I tutaj pojawia się mały problem, metody optymalizacji jednowymiarowej wymagają podania przedziału, w którym może znajdować się minimum. Może zdarzyć się tak, że dla dużych d, źle dobrany przedział poszukiwań $[a, b]$ sprawi, że krok $h^{(i)}$ będzie bardzo duży i nie trafimy w minimum. Co może się zdarzyć dla bardzo "płaskich" funkcji.

Dokładnie na taki problem natrafiłem podczas pracy. Funkcja w okolicach minimum posiadała bardzo duży gradient i w następnej iteracji x znacząco oddalał się od minimum.

Algorytm był zbieżny wyłącznie dla minimum, oddalenie się o 0.01 powodowało rozbieżność.

Próba rozwiązania problemu

Pierwszą rzeczą, której spróbowała najoczywistsza rzecz tj. ograniczenie gradientu. W przypadku gdy stawał się zbyt duży, dzieliłem go przez jego długość. Sprawdziło się to wyłącznie w początkowych iteracjach, ponieważ gdyż w przypadku gdy krok $h^{(i)}$ otrzymany przez metodę optymalizacji jednowymiarowej był zbyt duży, nadal oddalałem się od minimum.

Rozwiązanie problemu

Zmiana gradientu okazała się nietrafionym pomysłem, ponieważ przy okazji zmieniał się parametr β. Zamiast tego zmieniłem przedział poszukiwań optymalnego kroku, ponieważ to on ostatecznie był dodawany do x. Idelnym przedziałem okazał się

$$[a, b] = \left[max \left\{ \frac{-1}{|d|}, -|d| \right\}, min \left\{ \frac{1}{|d|}, |d| \right\} \right]$$

W przypadku bardzo dużych d, przedział poszukiwań był bardzo zawężany, co sprawiało, że krok także się zmiejszał.

Próbowałem także zmienić algorytm optymailzacji jednowymiarowej i dla podanej funkcji jedynym, który zadziałał był algorytm golden. Z algorytmem Fibonacciego już w drugiej iteracji (dla tych samych parametrów) x stawał się duży, a algorytm nie był zbieżny.

Algorytm golden

```
golden <- function(f, lower, upper, tol) {
  ratio <- 2 / (3 + sqrt(5))
  x1 <- (1 - ratio) * lower + ratio * upper
  f.x1 <- f(x1)
  while (abs(upper - lower) > 2 * tol) {
    x2 <- (1 - ratio) * x1 + ratio * upper
    f.x2 <- f(x2)
    if (f.x1 < f.x2) {
      upper <- lower
      lower <- x2
    } else {
      lower <- x1
      x1 <- x2
      f.x1 <- f.x2
    }
  }
```

Metoda gradientów sprzężonych Polaka-Ribiere'a - kod w R

```
library(numDeriv)
polak.ribiere <- function(f, x, tol) {
  beta <- 1; i <- 1;
  d <- -grad(f, x,)
  repeat {
    g <- function(a) {f(x + a * d)}
    d.magnitude <- sqrt(sum(d^2));
    grad.x <- grad(f, x)
    step <- golden(g,
                max(c(-1/d.magnitude,-d.magnitude)),
                min(c(1/d.magnitude,d.magnitude)), tol)
    new.x <- x + step * d
    grad.new.x <- grad(f, new.x)
    cat("(",i,")", "x=", x,"step=",step, "new.x=", new.x, "grad.x=",
        grad.x, "\n(",i,")", "grad.new.x=", grad.new.x, "d=", d, "\n");
    if (dist(rbind(new.x,x)) < tol) {
      return(new.x)
    }
    beta <- t(grad.new.x) %*%
              (grad.new.x - grad.x) /
              (t(grad.x) %*% grad.x)
    d <- -grad(f,new.x) + as.vector(beta) * d;
    x <- new.x; i <- i + 1;
  }
}
```

Rozwiązanie zadania

```
fn <- function(x) f(x[1], x[2]);
m <- polak.ribiere(fn, c(-0.5, -0.5), 1e-3)
```

```
## ( 1 ) x= -0.5 -0.5 step= 0.004422321 new.x= -0.5663348 -0.4004978 grad.x= 15 -22.5
## ( 1 ) grad.new.x= 31.14325 19.24268 d= -15 22.5
## ( 2 ) x= -0.5663348 -0.4004978 step= 0.00111821 new.x= -0.631116 -0.3770805 grad.x= 31.14325 19.24268
## ( 2 ) grad.new.x= -17.78508 10.04248 d= -57.93287 20.94176
## ( 3 ) x= -0.631116 -0.3770805 step= -0.001646764 new.x= -0.6050355 -0.3805576 grad.x= -17.78508 10.0424
## ( 3 ) grad.new.x= 6.04749 19.4507 d= -15.83741 2.111483
## ( 4 ) x= -0.6050355 -0.3805576 step= 0.0003170421 new.x= -0.6108902 -0.3861993 grad.x= 6.04749 19.4507
## ( 4 ) grad.new.x= -2.596436 9.747444 d= -18.46667 -17.79495
## ( 5 ) x= -0.6108902 -0.3861993 step= 0.09671282 new.x= -0.04925789 -1.029674 grad.x= -2.596436 9.74744
## ( 5 ) grad.new.x= -12.53166 -8.918604 d= 5.807217 -6.653455
## ( 6 ) x= -0.04925789 -1.029674 step= 0.0002636252 new.x= -0.04157633 -1.032338 grad.x= -12.53166 -8.918
## ( 6 ) grad.new.x= -8.018823 -13.92914 d= 29.1382 -10.10787
## ( 7 ) x= -0.04157633 -1.032338 step= 0.003018632 new.x= -0.004876797 -0.9946255 grad.x= -8.018823 -13.9
## ( 7 ) grad.new.x= -3.580418 5.535574 d= 12.15767 12.4934
## ( 8 ) x= -0.004876797 -0.9946255 step= 0.001019029 new.x= 0.003177166 -0.9957393 grad.x= -3.580418 5.53
## ( 8 ) grad.new.x= 0.05488171 2.356969 d= 7.903569 -1.093039
## ( 9 ) x= 0.003177166 -0.9957393 step= 0.001951316 new.x= 0.0004823992 -0.9999806 grad.x= 0.05488171 2.3
## ( 9 ) grad.new.x= 0.1968787 -0.1297763 d= -1.380999 -2.173571
## ( 10 ) x= 0.0004823992 -0.9999806 step= 0.001764625 new.x= -1.876585e-05 -0.9999936 grad.x= 0.1968787 -
## ( 10 ) grad.new.x= -0.009753898 0.01062207 d= -0.2840066 -0.007355346
```

Rozwiązanie zadania - sprawdzenie

Wynik ten pokrywa się z rozwiązaniem podawanym przez WolframAlpha

Input interpretation

minimize	$(1+(x+y+1)^2(19-14+3x^2-14y+6xy+3y^2))$ $(30+(2x-3y)^2(18-32x+12x^2+48y-36xy+27y^2))$

Global minima

(no global minima found)

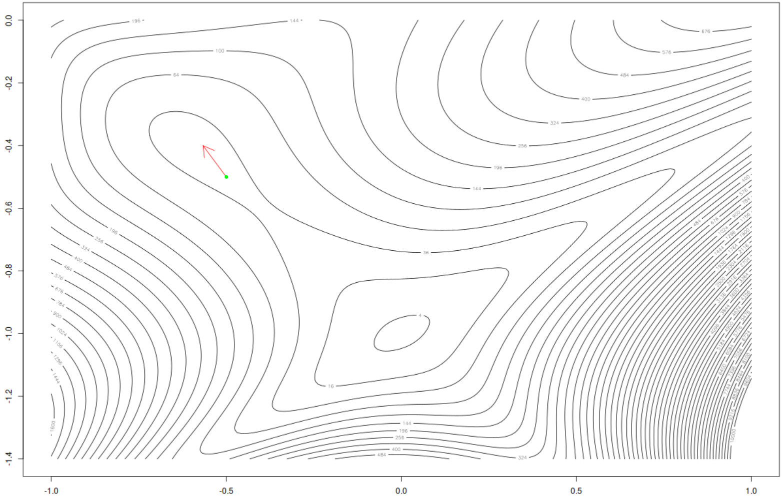
Local minima | Approximate form

$$\min\{(1+(x+y+1)^2(19-14+3x^2-14y+6xy+3y^2))\,(30+(2x-3y)^2(18-32x+12x^2+48y-36xy+27y^2))\} = 30 \text{ at } (x,y)=\left(-\frac{3}{5},-\frac{2}{5}\right)$$

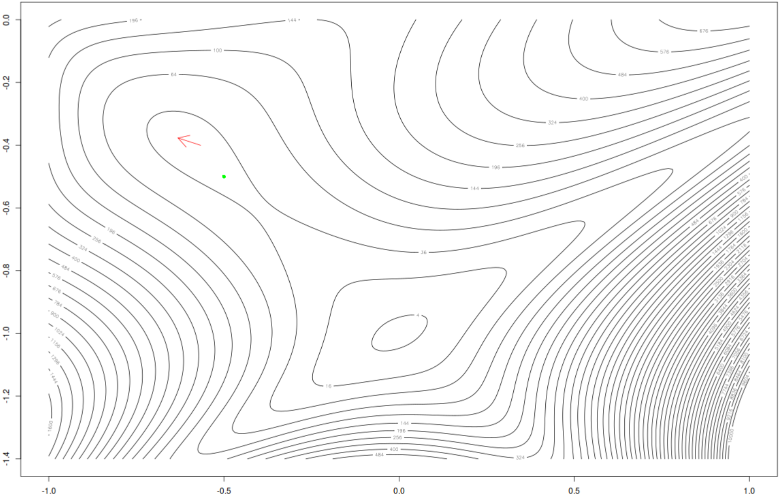
$$\min\{(1+(x+y+1)^2(19-14+3x^2-14y+6xy+3y^2))\,(30+(2x-3y)^2(18-32x+12x^2+48y-36xy+27y^2))\} = 3 \text{ at } (x,y)=(0,-1)$$

Rysunek 2: Wynik podany przez Wolframa

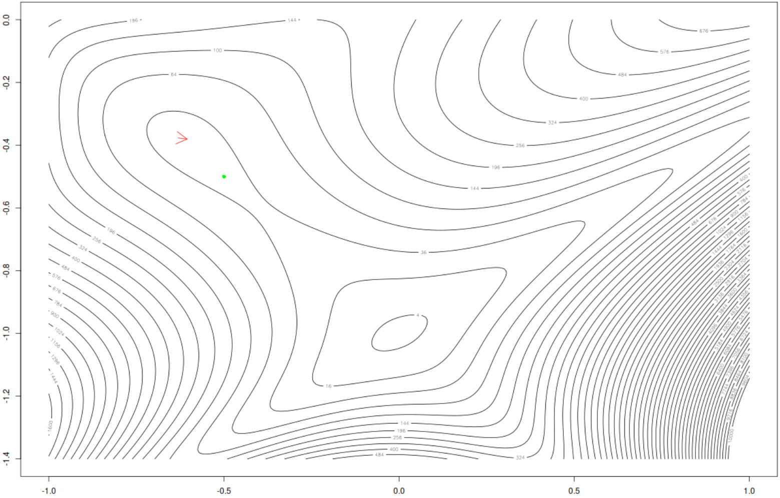
Rozwiązanie - wizualizacja



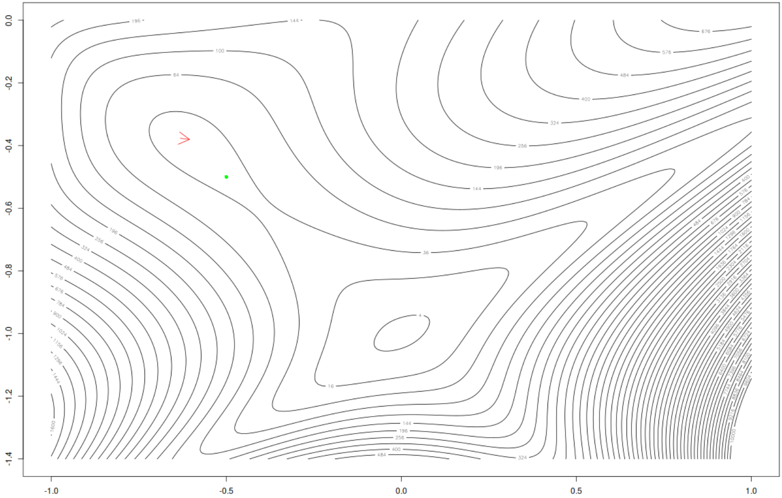
Rozwiązanie - wizualizacja



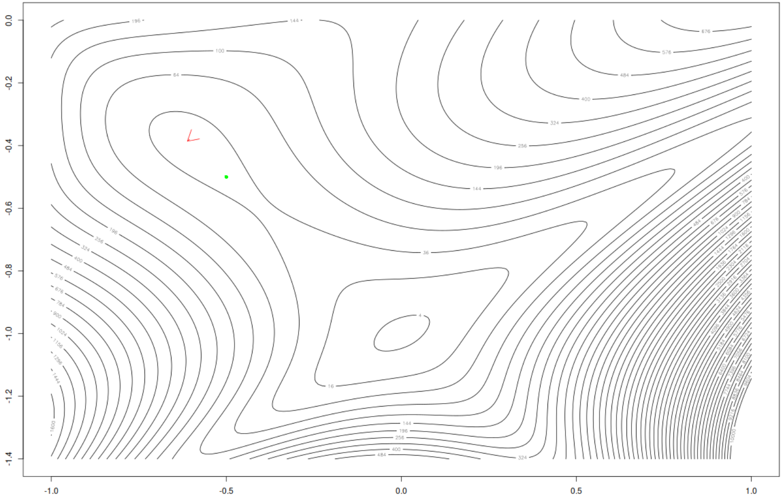
Rozwiązanie - wizualizacja



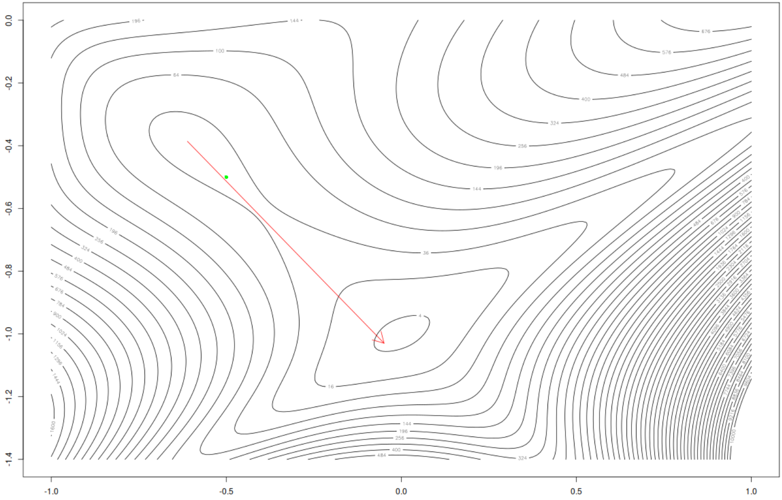
Rozwiązanie - wizualizacja



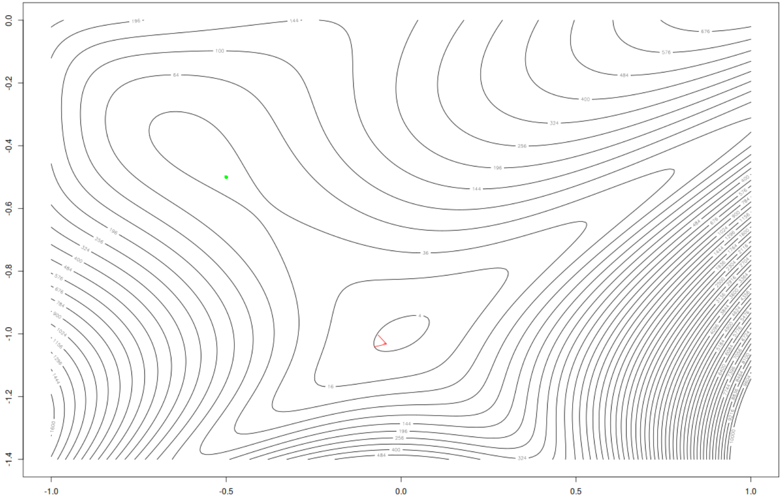
Rozwiązanie - wizualizacja



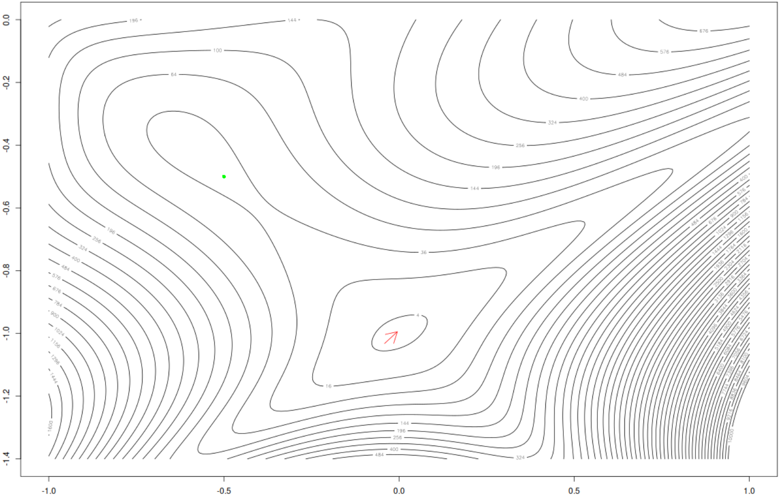
Rozwiązanie - wizualizacja



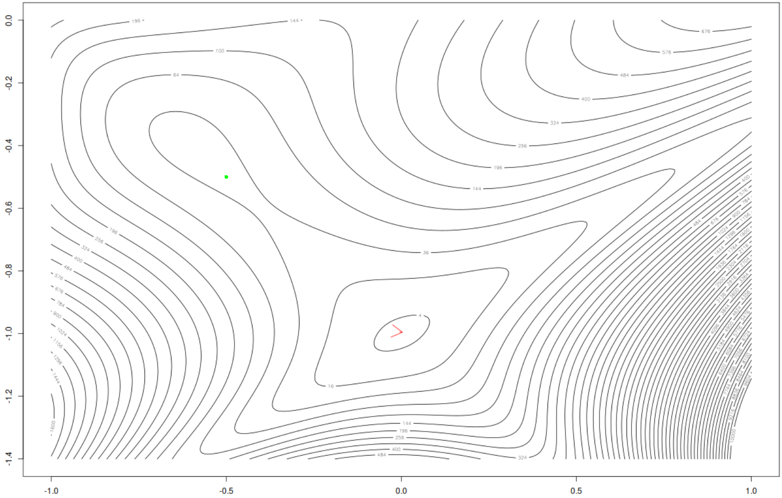
Rozwiązanie - wizualizacja



Rozwiązanie - wizualizacja



Rozwiązanie - wizualizacja



Rozwiązanie - wizualizacja