

# Rozwiązanie zadania N12

Krzysztof Waniak

Znaleźć wszystkie rozwiązania układu równań:

$$\begin{cases} \sin(x + y^2 + 1) = 0, \\ xy - 1 = 0 \end{cases}$$

w kwadracie  $(-2,2) \times (-2,2)$ . Można korzystać z gotowych bibliotek numerycznych, ale nie z gotowych programów typu Mathematica.

Wiadomo nam, że układ ma jedno rozwiązanie (źródło: WolframAlpha) w wyznaczonym przedziale. Tak, więc trzeba je iteracyjnie znaleźć w zadanym kwadracie. Do rozwiązania powyższego układu równań wykorzystano funkcje biblioteczne GSL (szukanie miejsc zerowych wielu zmiennych).

Kod programu:

```
#include<stdlib.h>
#include<stdio.h>
#include<gsl/gsl_vector.h>
#include<gsl/gsl_multiroots.h>
#include<ctype.h>          /* zawiera F_OK itp.    */
#include<unistd.h>         /* zawiera funkcje access(), usleep() */
#define wyp(a) printf(("#a "\n")
#define wyp2(a) printf(("#a
#define wypisz(a) printf("%.3f",a)
#define wypisz1(a) printf("%3u",a)
#define wypisz2(a) printf("%.13f",a)
#define karetka printf("\n")
#define karetka2 printf("\n\n")
#define space printf(" ")
#define space2 printf("  ")

void Wynik(gsl_multiroot_fsolver * a)
{
    wyp2(Rozwiazanie: x =);
    space;
    wypisz(gsl_vector_get(a -> x, 0));
    wyp2(: y =);
    space;
    wypisz(gsl_vector_get(a -> x, 1));
    karetka2;
}

void wypisywanie(size_t krok, gsl_multiroot_fsolver * a)
{
    wyp2(Krok =);
    wypisz1(krok+1);
    space2;
    wyp2(x =);
    space;
    wypisz(gsl_vector_get(a -> x, 0));
    space2;
    wyp2(y =);
```

```

space;
wypisz(gsl_vector_get(a -> x, 1));
karetka;

wyp2(f1(x) =);
space;
wypisz2(gsl_vector_get(a -> f, 0));
space;
space2;
wyp2(f2(x) =);
space;
wypisz2(gsl_vector_get(a -> f, 1));
karetka;
}
/* dokladnosc do ok. 10^-8 - 10^-10*/
int zero(const gsl_vector * x, void *params, gsl_vector * f)
{
    const double x0 = gsl_vector_get(x, 0);
    const double x1 = gsl_vector_get(x, 1);
    const double y0 = sin(x0 + (x1 * x1) + 1);
    const double y1 = x0 * x1 - 1;

    gsl_vector_set(f, 0, y0);
    gsl_vector_set(f, 1, y1);

    return GSL_SUCCESS;
}

int main(void)
{
    const gsl_multiroot_fsolver_type *T;
    gsl_multiroot_fsolver *a;

    int status;
    size_t krok = 0;

    const size_t n = 2;
    gsl_multiroot_function f = { &zero, n };

    double x_init[2] = { -2.0, -2.0 };
    gsl_vector *x = gsl_vector_alloc(n);

    gsl_vector_set(x, 0, x_init[0]);
    gsl_vector_set(x, 1, x_init[1]);

    T = gsl_multiroot_fsolver_hybrids;
    a = gsl_multiroot_fsolver_alloc(T, 2);
    gsl_multiroot_fsolver_set(a, &f, x);

    wypisywanie(krok, a);

    do
    {
        krok++;
        status = gsl_multiroot_fsolver_iterate(a);
        wypisywanie(krok, a);

        if (status)
        {
            break;
        }
    }

```

```

        status = gsl_multiroot_test_residual(a -> f, 1e-8);
    } while (status == GSL_CONTINUE && krok < 1000);

    /* printf("%s\n", gsl_strerror(status)); */
    Wynik(a);

    gsl_multiroot_fsolver_free(a);
    gsl_vector_free(x);

    /*system("pause");*/
    return 0;
}

```

Wynik działania programu:

```

Krok = 1  x = -2.000  y = -2.000
f1(x) = 0.1411200080599  f2(x) = 3.0000000000000
Krok = 2  x = -0.771  y = -1.729
f1(x) = -0.0745917234606  f2(x) = 0.3335288452854
Krok = 3  x = -0.639  y = -1.674
f1(x) = -0.0225250648752  f2(x) = 0.0690663267053
Krok = 4  x = -0.606  y = -1.658
f1(x) = -0.0020064880585  f2(x) = 0.0040063341190
Krok = 5  x = -0.604  y = -1.657
f1(x) = -0.0000998246929  f2(x) = 0.0001602292179
Krok = 6  x = -0.604  y = -1.657
f1(x) = -0.0000021960523  f2(x) = 0.0000033112533
Krok = 7  x = -0.604  y = -1.657
f1(x) = -0.0000000137441  f2(x) = 0.0000000206628
Krok = 8  x = -0.604  y = -1.657
f1(x) = -0.0000000000040  f2(x) = 0.0000000000061
Rozwiązanie: x = -0.604: y = -1.657

Aby kontynuować, naciśnij dowolny klawisz . . . _

```