# Optimal balanced circular packing problem

Romanova Tatiana[1], Stetsyuk Petro[2], Kovalenko Anna[1]
*stetsyukp@gmail.com*

[1]A.N. Podgorny Institute for Mechanical Engineering Problems, Kharkov, Ukraine
[2]V.M. Glushkov Institute of Cybernetics, Kiev, Ukraine

OPTIMA 2014, Petrovac, Montenegro
September 28 – October 4

# Outline

# Outline

# Optimization problem

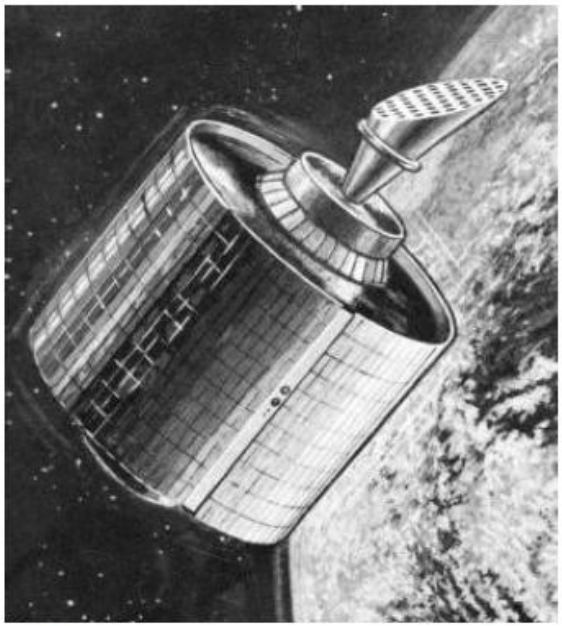
Let a collection of circles $S_i$, their radii $r_i$ and weights $w_i$, $i = 1, \ldots, m$, be given. We assume that the gravity centre of circle $S_i$ is placed at its centre.

## Optimization problem

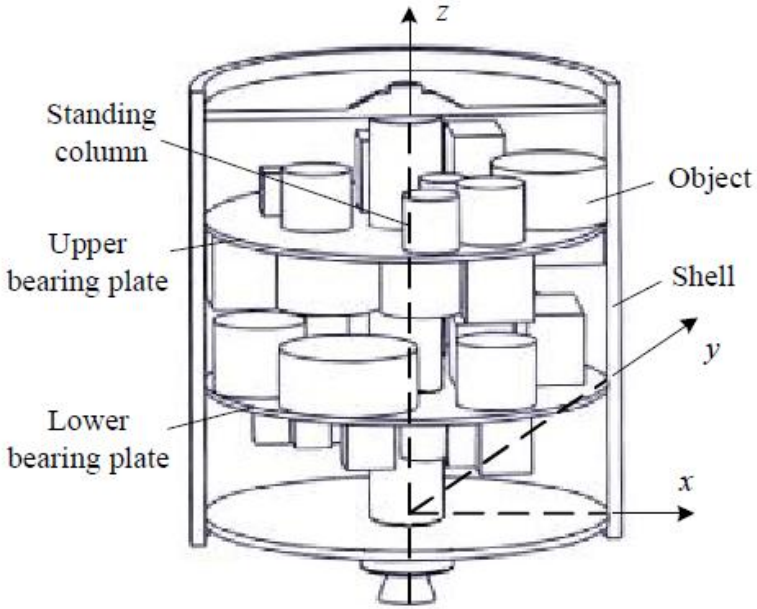
Pack the collection of circles $S_i$, $i = 1, \ldots, m$, into containing circle $S$ of minimal radius, provided that the gravity centre of the collection of circles is located at the centre of the circle $S$ which coinsides with the origin.

# Application 1

for satellite modeling (space engineering)



a. the international communication satellite

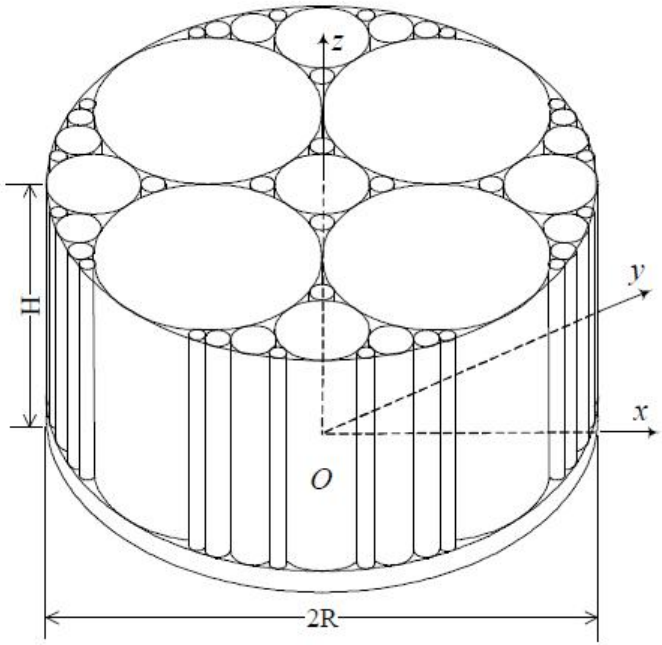


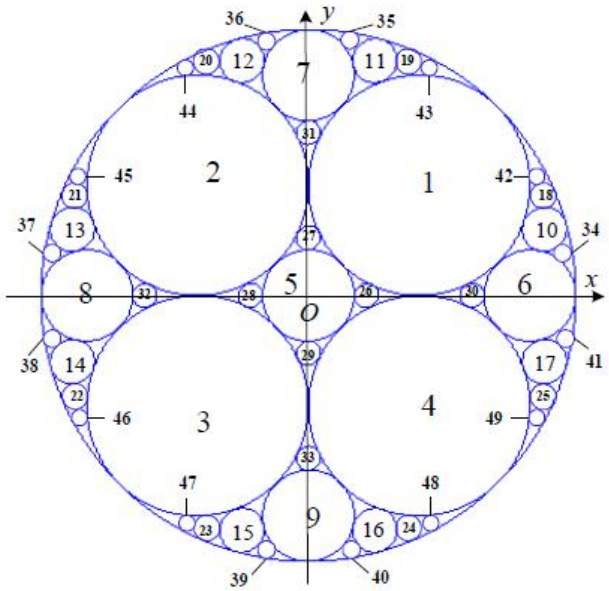

b. the simplified satellite module

FASANO G., PINTER J.D., EDS., Modeling and Optimization in Space Engineering. Springer Optimization and Its Applications. – Springer. – New York. – 2013. – 404 p.

# Application 2

Cylinders and a container have the same height




Che C., Wang Y., Teng H., Test problems for quasi-satellite packing: Cylinders packing with behavior constraints and all the optimal solutions known, Optimization Online (2008).

# Outline

# Notation

## Variables

We denote a variable centre of circle $S_i$ by $(x_i, y_i)$ and a variable radius of containing circle $S$ by $r$.

## Parameters

Let $\lambda_i = w_i / \sum_{i=1}^{m} w_i$, $i = 1, \ldots, m$, and $r_{low} = \max\limits_{i} r_i$ be the trivial lower bound of $r$.

# Nonlinear programming problem

find

$$r^* = \min_{x,y,r} r \tag{1}$$

subject to

$$x_i^2 + y_i^2 \leq (r - r_i)^2, \quad i = 1, \ldots, m, \tag{2}$$

$$(x_i - x_j)^2 + (y_i - y_j)^2 \geq (r_i + r_j)^2, \quad 1 \leq i < j \leq m, \tag{3}$$

$$\sum_{i=1}^{m} \lambda_i x_i = 0, \quad \sum_{i=1}^{m} \lambda_i y_i = 0, \tag{4}$$

$$r \geq r_{low}, \tag{5}$$

where $x = (x_1, \ldots, x_m)$, $y = (y_1, \ldots, y_m)$.

# Constraints (2) - (5) mean

$$\varphi_i(x, y, r) = x_i^2 + y_i^2 - (r - r_i)^2 \leq 0, \quad i = 1, \ldots, m, \qquad (2')$$

i.e. circles $S_i$, $i = 1, \ldots, m$ are contained into the containing circle $S$, if constraint (5) is satisfied

$$\phi_{ij}(x, y) = -(x_i - x_j)^2 - (y_i - y_j)^2 + (r_i + r_j)^2 \leq 0,\ 1 \leq i < j \leq m, \quad (3')$$

i.e. each pair of circles from the collection $S_i$, $i = 1, \ldots, m$ do not overlap,

$$\Phi_1(x, y) = \sum_{i=1}^{m} \lambda_i x_i = 0, \quad \Phi_2(x, y) = \sum_{i=1}^{m} \lambda_i y_i = 0, \qquad (4')$$

i.e. gravity center of the collection of circles $S_i$, $i = 1, \ldots, m$ is placed at the origin (the center of the containing circle $S$).

# Outline

# Step 1: we reduce the problem (1)–(5)

to an unconstrained nonsmooth minimization problem

$$\min_{x\in R^m, y\in R^m, r} F(x,y,r), \qquad (*)$$

where

$$F(x,y,r)=r+P_1\left(\sum_{i=1}^{m}\max\{0,\varphi_i(x,y,r)\}+\sum_{i=1}^{m}\sum_{j>i}^{m}\max\{0,\phi_{ij}(x,y)\}\right)$$

$$+P_2\sum_{k=1}^{2}\max\{0,-\Phi_k(x,y)-\varepsilon_k,\Phi_k(x,y)-\varepsilon_k\}+P_3\max\{0,-r+r_{low}\}.$$

Here $P_1, P_2, P_3$ are nonsmooth penalties (positive).

# Step 2

We use the combination

of Shor's $r$-algorithm and multistart method

to find "good" local extrema

for the problem (*).

# Outline

# The quadratic extremal problem

find

$$f^* = (r^*)^2 = \min_{r,x,y} r^2, \tag{6}$$

subject to

$$x_i^2 + y_i^2 - r^2 + 2r_i r - r_i^2 \le 0, \quad i = 1, \dots, m, \tag{7}$$

$$-x_i^2+2x_ix_j-x_j^2-y_i^2+2y_iy_j-y_j^2+(r_i+r_j)^2\le 0,\ 1\le i<j\le m, \tag{8}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_i\lambda_j x_i x_j = 0, \quad \sum_{i=1}^{m}\sum_{j=1}^{m}\lambda_i\lambda_j y_i y_j = 0, \tag{9}$$

$$r^2 - (r_{low} + r_{up})r + r_{low}r_{up} \le 0. \tag{10}$$

Here constraints (7) and (8) correspond to constraints (2) and (3), two equations in (9) correspond to equations (4). Quadratic inequality (10) follows from $r_{low} \le r \le r_{up}$, where $r_{up}$ may be found by the local search algorithm.

# Shor's dual bound $\psi^*$

gives a lower approximation of the objective function in the quadratic extremal problem.

The value of $\psi^*$ can be found with given accuracy by methods of minimization of nonsmooth convex functions.

 Shor N.Z. Nondifferentiable optimization and polynomial problems. – Kluwer Academic Publishers (1998)

# Properties of bound $\psi^*$

1) $\psi^* \leq f^* = (r^*)^2$ (Shor, 1985)

2) $r^* \geq \sqrt{\psi^*} \geq r_{low}$ (from problem (6)–(10))

It means, that bound $\sqrt{\psi^*}$ can be used to prove the global minimum for the problem (1)-(5).

# Outline

# Program ralgb5

For computational experiments the program ralgb5 was used.

This program realizes

Shor's $r(\alpha)$-algorithm with adaptive step-size regulation.

# Example 1 ($m = 5$)

Circles have different radii and weights:

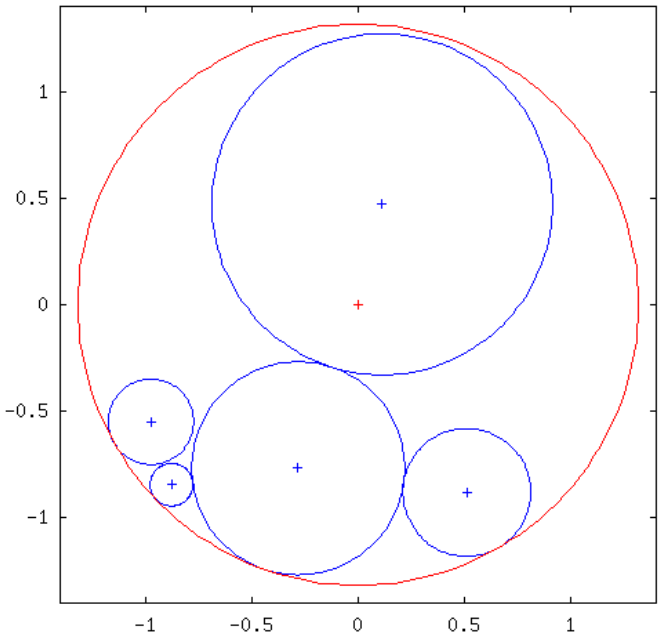
- $r_1 = 0.1$, $r_2 = 0.2$, $r_3 = 0.3$, $r_4 = 0.5$, $r_5 = 0.8$
- $w_1 = 0.0785$, $w_2 = 0.314$, $w_3 = 0.7065$, $w_4 = 1.9625$, $w_5 = 5.024$

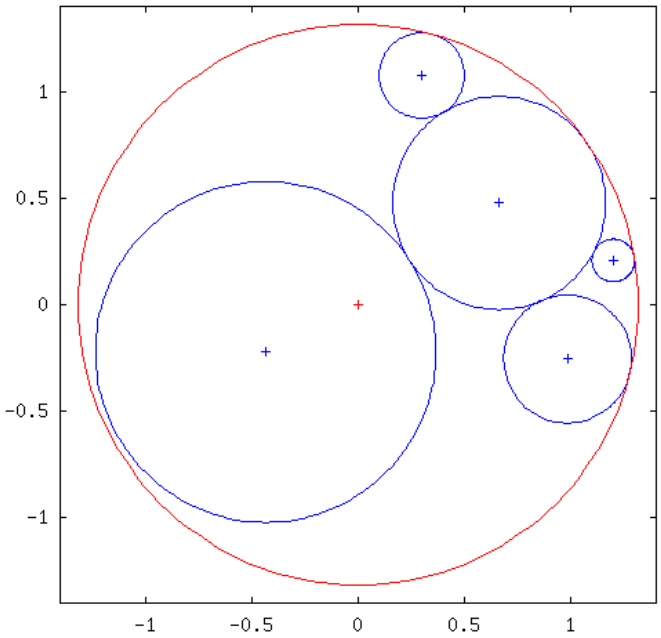
Tolerance for the components of the center of gravity

- $\varepsilon_1 = \varepsilon_2 = 0.0001$

Nenakhov E.I., Romanova T.E., Stetsyuk P.I., Balanced packing problem of circles in a circle of minimum radius, Theory of optimal solutions, 143–153 (2013).

# Two close local minima




$r_{up} = 1.3175 \neq r^*$

$r_{up} = 1.3161 = r^*$ ???

# The proof that $r^* = 1.3161$

If

$$r_{low} = 0.8, \quad r_{up} = 1.35 > r^* = 1.3161$$

then

$$\psi^* = 1.7309$$

and we have

$$r^* \geq \sqrt{\psi^*} \geq 1.3156$$

Stetsyuk P.I., Romanova T.E., Schiethauer G.
On the global minimum of the objective function in a balanced circular packing problem, Dopovidi NAN Ukraine, 6, 53–57 (2014).

# Example 2 ($m = 7$)

Six circles have the same radius and weight:

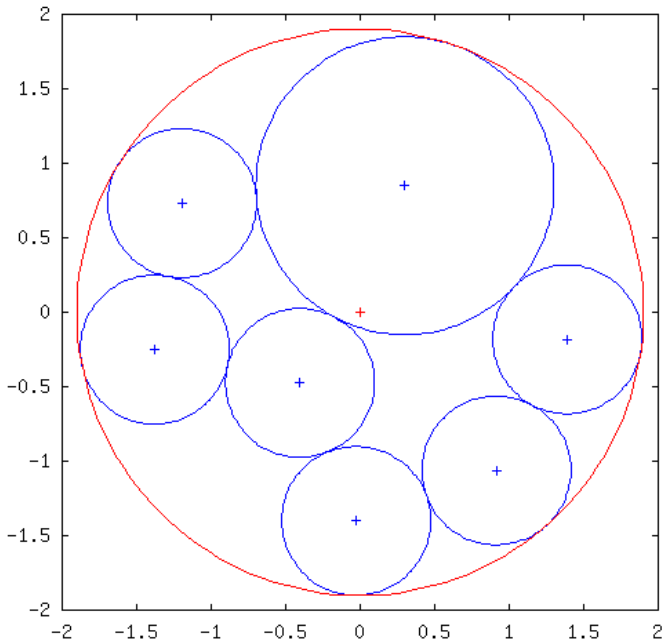
- $r_1 = 1,\ r_2 = r_3 = r_4 = r_5 = r_6 = r_7 = 0.5$
- $m_1 = 4,\ m_2 = m_3 = m_4 = m_5 = m_6 = m_7 = 1$

Tolerance for the components of the center of gravity

- $\varepsilon_1 = \varepsilon_2 = 0.0001$

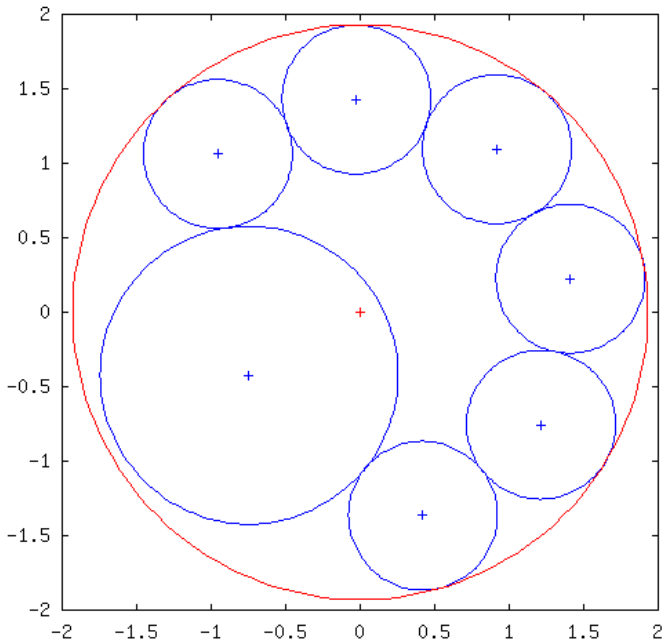
Pshenichnyi B.N., Sobolenko L.A., Linearization method for inverse convex programming. Cybernetics and Systems Analysis, 31(6), 852-862 (1995).

# Optimal placement of circles




without balance constraints

$r_{up} = 1.9012 = r^*$




with balance constraints

$r_{up} = 1.9289 = r^*$ ???

# There is no proof that $r^* = 1.9289$

If

$$r_{low} = 1.0, \quad r_{up} = 1.9290 > r^* = 1.9289$$

then

$$\psi^* = 2.4299$$

and we have

$$r^* \geq \sqrt{\psi^*} \geq 1.5588$$

# Conclusion

Dual bound $\psi^*$ can be successively improved by adding superfluous quadratic constraints to (7)–(10).
These constraints are non-trivial consequences of problem conditions. For instance, similar to (10) we can construct additional constraints for some of variables $x_i$, $y_i$, $i = 1, \ldots, m$, using ranges of their variations.

# Questions?

THANK YOU
FOR YOUR ATTENTION!

# BACK UP SLIDES: Comments for program ralgb5

```
#  Octave-code ralgb5 realizes Shor's $r(\alpha)$-algorithm
#  with adaptive step for minimizing of convex function f(x).
#  ralgb5 uses user-defined octave's function [f,g]=calcfg(x),
#  which calculate $f=f(x)$ and subgradient $g(x)$ at $x$.
#  Input parameters:
#    calcfg - name of function for f and g calculation
#    x  -  starting point x0(1:n) (is erased at the return)
#    alpha - the space dilation coefficient
#    h0, nh, q1, q2 - adaptive step parameters
#    epsx, epsg, maxitn  - stop parameters
#  Output parameters:
#    xr -  the found minimum point xr(1:n)
#    fr -  the function value at minimum point
#    itn - number of iterations
#    ncalls - number of calcfg calls
#    istop - the stop code (2=epsg,3=epsx,4=maxitn,5=error)
```

# BACK UP SLIDES: Octave-function ralgb5

```
function [xr,fr,itn,ncalls,istop]=ralgb5(calcfg,x,alpha,h0,q1, # row001
                                         q2,nh,epsg,epsx,maxitn);
itn=0; hs=h0; B=eye(length(x)); xr=x;                          # row002
ncalls = 1; [fr,g0] = calcfg(xr);                              # row003
printf("itn %4d f %14.6e fr %14.6e ls %2d ncalls %4d\n",       # row004
       itn, fr, fr, 0, ncalls);
if(norm(g0) < epsg) istop = 2;  return; endif                  # row005
for (itn = 1:maxitn)                                           # row006
   dx = B * (g1 = B' * g0)/norm(g1);                           # row007
   d = 1; ls = 0; ddx = 0;                                     # row008
   while (d > 0)                                               # row009
      x -= hs * dx; ddx += hs * norm(dx);                      # row010
      ncalls ++; [f, g1] = calcfg(x);                          # row011
      if (f < fr)  fr = f; xr = x; endif                       # row012
      if(norm(g1) < epsg) istop = 2;  return; endif            # row013
      ls ++; (mod(ls,nh)==0) && (hs *= q2);                    # row014
      if(ls > 500) istop = 5;  return; endif                   # row015
      d = dx' * g1;                                            # row016
  endwhile                                                     # row017
  (ls == 1) && (hs *= q1);                                     # row018
  printf("itn %4d f %14.6e fr %14.6e ls %2d ncalls %4d\n",     # row019
         itn, f, fr, ls, ncalls);
  if(ddx < epsx) istop = 3;  return; endif                     # row020
  xi = (dg = B' * (g1 - g0) )/norm(dg);                        # row021
  B += (1 / alpha - 1) * B * xi * xi';                         # row022
  g0 = g1;                                                     # row023
endfor                                                         # row024
istop = 4;                                                     # row025
endfunction                                                    # row026
```

# BACK UP SLIDES: The parameters for ralgb5

For minimization of nonsmooth functions we recommend:

$$\alpha = 2 \div 3,\ h_0 = 1.0,\ q_1 = 1.0,\ q_2 = 1.1 \div 1.2,\ n_h = 2 \div 3.$$

If the bound on distance from starting point $x_0$ to the minimum point $x^*$ is given, then it is reasonable to choose initial step $h_0$ to be approximately equal to $\|x_0 - x^*\|$.
For minimization of smooth functions we recommend:

$$q_1 = 0.8 \div 0.95.$$

With this choice of parameters, as a rule, the number of descents in a direction rarely exceeds two, and for the $n$ iterations the accuracy of the values of function is improved from three to five times.