

Stanislav Bartoň

Three dimensional modelling of the peach in MAPLE

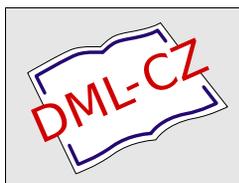
In: Jan Chleboun and Petr Příkryl and Karel Segeth and Tomáš Vejchodský (eds.): Programs and Algorithms of Numerical Mathematics, Proceedings of Seminar. Dolní Maxov, June 1-6, 2008. Institute of Mathematics AS CR, Prague, 2008. pp. 7–14.

Persistent URL: <http://dml.cz/dmlcz/702851>

**Terms of use:**

© Institute of Mathematics AS CR, 2008

Institute of Mathematics of the Czech Academy of Sciences provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use*.



This document has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library*  
<http://dml.cz>

# THREE DIMENSIONAL MODELLING OF THE PEACH IN MAPLE\*

Stanislav Bartoň

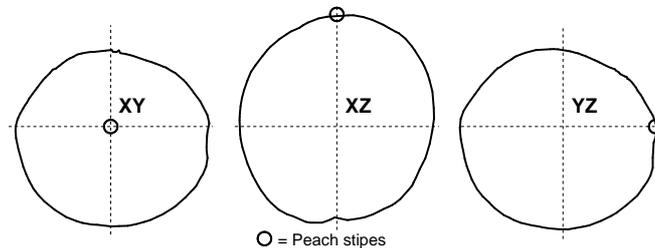
## Abstract

Linearized Gauss-Newton iteration method is used to determine main axes of the three-dimensional ellipsoid approximating a peach. Three independent photos displaying the peach as ground, side, and front view are used as data sources. System MAPLE 11 was used as a computer environment. A practical example is presented in order to demonstrate the usage of all required commands. The quality of approximation is evaluated as a final part of the paper.

## 1. Introduction

The knowledge of the object's shape is usually one of the critical parameters for evaluating its mechanical properties. Concerning peaches, the proportions are significant e.g. for storage processing, deep freezing namely. Since only undamaged and healthy fruits can be processed, it is important to determine maximum levels of mechanical loading, thus to determine the modulus of elasticity [1]. The modulus of elasticity can be determined by many nondestructive methods. One of them is based on measuring of the acoustic wave propagation [2].

The most precise method of the object dimensions measuring is 3D scanning [3], but this method is very expensive, demands a lot of time and manual work, and finally produces huge amount of data. Digging the main shape parameters represents rather complicated problem. This article shows the procedure of the approximation of the peach shape by triaxial ellipsoid and using of three digital photos. These photos must be taken in mutually perpendicular orientations, see Figure 1.



**Fig. 1:** *Basic peach photos.*

---

\*This research has been supported by the Grant Agency of the Czech Academy of Sciences under Contract No. IAA201990701, *Behaviour of selected agricultural products under impact loading.*

There are many computer codes intended for the extraction of pixels coordinates from the objects perimeter. The approach described in [4] was used in this paper. Coordinates of the perimeter's pixels were saved in three independent files. Used algorithm reduces the number of pixels and orders them in such way, that resulting polygon approximates the perimeter with the accuracy better than  $\pm 1$  pixel. This method reduces the number of pixels from thousands into hundreds of polygon vertices without loss of accuracy.

## 2. Coordinate system

Let us assume that a peach will be approximated by a triaxial ellipsoid with the axis  $A$ ,  $B$ , and  $C$  in the spherical coordinate system. If the peach is properly oriented, see Figure 1, photos can be interpreted as equatorial cross section -  $XY$ , and cut along  $0^\circ$  meridian cross section -  $XZ$  and  $90^\circ$  meridian -  $YZ$ . The cross section curves may be approximated by mutually dependent ellipses. Their semiaxes will be  $A$ ,  $B$  for  $XY$ ,  $A$ ,  $C$  for  $XZ$  and  $B$ ,  $C$  for  $YZ$ . It must be assumed that these ellipses will have different centers and may be rotated by a small angle around different centers. The following description will be used:  $[Ax, Ay]$  = temporary unknown center of the ellipse approximating the  $XY$  cut,  $Ra$  - rotation angle of the approximating ellipse around center. Variables  $[Bx, Bz]$ ,  $Rb$ ,  $[Cy, Cz]$  and  $Rc$  have the same meaning for cross sections  $XZ$  and  $YZ$ . Parametric shape will be used to describe all three ellipses. The first ellipse will be described by the function  $[A \cos(fa), B \sin(fa)]$ , the second one by  $[A \cos(fb), C \sin(fb)]$  and the third one by  $[B \cos(fc), C \sin(fc)]$ , where  $fa$ ,  $fb$ ,  $fc$  are mutually independent parameters,  $-\pi \leq fa, fb, fc \leq \pi$ .  $[X, Y]$ ,  $[Y, Z]$  and  $[X, Z]$  are coordinates of the polygon vertices, saved in three different files. Using these variables we can determine the difference between coordinates of the general vertex point and the nearest corresponding point on the ellipse. We shall use the least squares method (LSQ) to compute all unknown variables which will minimize  $Qxy + Qxz + Qyz$ , where:

$$Qxy = \sum_{i=1}^{N_{xy}} (A \cos(fa_i) - (X_i - Ax) \cos(Ra) - (Y_i - Ay) \sin(Ra))^2 + (B \sin(fa_i) - (X_i - Ax) \sin(Ra) - (Y_i - Ay) \cos(Ra))^2, \quad (1)$$

$[X_i, Y_i]$  are coordinates of the polygon vertices of the  $XY$  cut and  $fa_i$  are parameters of the points laying on the ellipse approximating the polygon.  $Qxz$  and  $Qyz$  are defined very similarly.

## 3. Preparation of the iteration

Equation (1) is non-linear for all unknowns. It means that an iteration method based on the linearization must be used. One of the best approaches derived by Newton and improved by Gauss is the well-known Gauss-Newton method. Purpose of the presented article is not to explain this method, but to show how to use it on a complicated and large problem. To handle matrices, the MAPLE library "Lin-

earAlgebra” is used. The library “plots” simplifies the visualization of results. The approach presented in this article is as generalization of [5] and [6].

```
> restart; with(LinearAlgebra): with(plots):
```

In order to limit the scope, only commands and variables describing the coordinate difference between a vertex and the corresponding point on the ellipse for  $XY$  cross section will be described. Other MAPLE commands are very similar.

```
> xa:=A*cos(fa)-((X-Ax)*cos(Ra)+(Y-Ay)*sin(Ra)):
> ya:=B*sin(fa)+((X-Ax)*sin(Ra)-(Y-Ay)*cos(Ra)):
> # similarly xb, zb, yc, zc
```

The difference between origin of the series expansion and the actual variable value will be substituted as a new variable. This variable has  $D$  at the beginning of its name. The value of this variable will represent the correction for the corresponding variable during the iteration process.

```
> Dsu:=map(u->u-cat(u||0)=cat(D||u),Var);
> Mxa:=subs(Dsu,Mxa); # similarly Mya, Mxb, Mzc, Myc, Mzc
```

$$\begin{aligned} Mxa := & A0 \cos(fa0) - (Y - Ay0) \sin(Ra0) - (X - Ax0) \cos(Ra0) + DA \cos(fa0) \\ & - A0 \sin(fa0) Dfa + DAx \cos(Ra0) + DAy \sin(Ra0) \\ & + ((X - Ax0) \sin(Ra0) - (Y - Ay0) \cos(Ra0)) DRa \end{aligned}$$

Complications arise with the parameters. For each vertex/point the best possible initial approximation is needed. If we have  $N$  points, we have to find  $N$  corresponding initial parameters. That is why correction for parameters from the Taylor series must be separated. Now it is possible to create a lists containing terms multiplying the individual differences.

```
> JFxa:=select(has,Mxa,Dfa)/Dfa: # similarly JFya ... JFzc
> JXa:=map(u->select(has,Mxa,u)/u,map(u->rhs(u),Dsu[4..-1]));
> # similarly JYa ... JZc
```

$$JXa := [\cos(fa0), 0, 0 \cos(Ra0), 0, 0, 0, 0, (X - Ax0) \sin(Ra0) - (Y - Ay0) \cos(Ra0), 0, 0]$$

Finally, separation of free terms is necessary.

```
> RXa:=remove(has,Mxa,map(u->rhs(u),Dsu)); # similarly RYa .. RZc
```

$$RXa := A0 \cos(fa0) - (Y - Ay0) \sin(Ra0) - (X - Ax0) \cos(Ra0)$$

#### 4. Initial values iteration

At this point, the files containing coordinates of vertices can be read. The whole polygon will be moved, so its mass center will be in the center of the coordinate system.  $Na$ ,  $Nb$ ,  $Nc$  represents the number of vertices in the individual cross sections and  $Ta$ ,  $Tb$ ,  $Tc$  are corresponding centers of their masses.

```
> read "007a_3.sav": Na:=nops(XYL):
> Ta:=sum(XYL[i],i=1..Na)/Na: XY:=map(u->u-Ta,XYL):
> # similarly 007b.sav --> Nb, Tb, XZ and 007c.sav --> Nc, Tc, YZ;
> 'Na'=Na, 'Nb'=Nb, 'Nc'=Nc, 'Ta'=Ta, 'Tb'=Tb, 'Tc'=Tc;
```

$$Na = 92, Nb = 80, Nc = 96,$$

$$Ta = [407.744, 331.126], Tb = [446.333, 324.705], Tc = [400.202, 363.912]$$

The initial values for axes may be determined as average of maximal diameters of the individual cross sections and arguments of the vertices in the polar coordinate system may be used as initial values of parameters. If photos were taken carefully the initial values for displacements and rotations may be zero.

```
> A0:=0.5*(max(map(u->u[1],XY)[])+max(map(u->u[1],XZ)[])):
> Fa:=map(u->op(2,polar(u[1]+I*u[2])),XY):
> # similarly XY, YZ --> B0; XZ, YZ --> C0; XZ --> Fb; YZ --> Fc
> Ax0:=0: Ay0:=0: Bx0:=0: Bz0:=0: Cy0:=0: Cz0:=0: Ra0:=0: Rb0:=0: Rc0:=0:
```

#### 5. Iteration

The most important variable in the iteration process is the Jacobi Matrix  $J$ . This matrix is composed from 24 submatrices, 12 of them are zero matrices, see construction of the Maple variable  $J$ . Its construction shows the Figure 2 and exactly corresponds with the structure of the command line for the variable  $J$ . Non zero elements are shown as crosses. It can be seen, that the Jacobi matrix is very sparse, partially filled up columns correspond to corrections of the variables  $[A, B, C, Ax, Ay, Bx, Bz, Cy, Cz, Ra, Rb, Rc]$ .

The vector of right sides contains variable  $R$  and vector  $DV$  is vector of corrections of initial values of searched variables. The variable  $eps$  controls whole iteration, which runs until  $eps$  is less than  $10^{-5}$ . The quadratic norm of the vector of corrections  $DV$  is assigned into  $eps$ . To reduce the big steps between corrections, corrections will be reduced to one half. Because at first few passes through loop corrections the parameters may be higher than  $2\pi$ , it is necessary to recompute all parameters into range  $< -\pi, \pi >$ . Subsequently, the corrections to all other variables are added.

```
> eps:=1: while eps>1e-5 do;
> J:=Matrix([[DiagonalMatrix(map(u->subs(fa0=u,JFxa),Fa)),ZeroMatrix(Na,Nb),
> ZeroMatrix(Na,Nc), Matrix(zip((u,v)->subs(fa0=u,X=v[1],Y=v[2],JXa),Fa,XY))],
> [DiagonalMatrix(map(u->subs(fa0=u,JFya),Fa)),ZeroMatrix(Na,Nb),
> ZeroMatrix(Na,Nc),Matrix(zip((u,v)->subs(fa0=u,X=v[1],Y=v[2],JYa),Fa,XY))],
> [ZeroMatrix(Nb,Na),DiagonalMatrix(map(u->subs(fb0=u,JFxb),Fb)),
> ZeroMatrix(Nb,Nc), Matrix(zip((u,v)->subs(fb0=u,X=v[1],Z=v[2],JXb),Fb,XZ))],
> [ZeroMatrix(Nb,Na),DiagonalMatrix(map(u->subs(fb0=u,JFzb),Fb)),
```

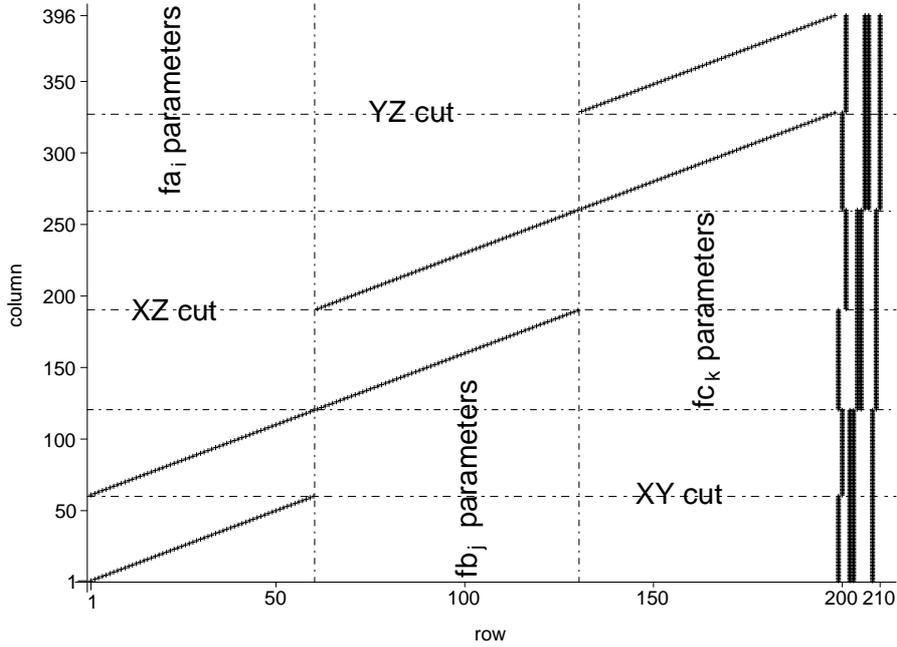


Fig. 2: Structure of the Jacobi matrix.

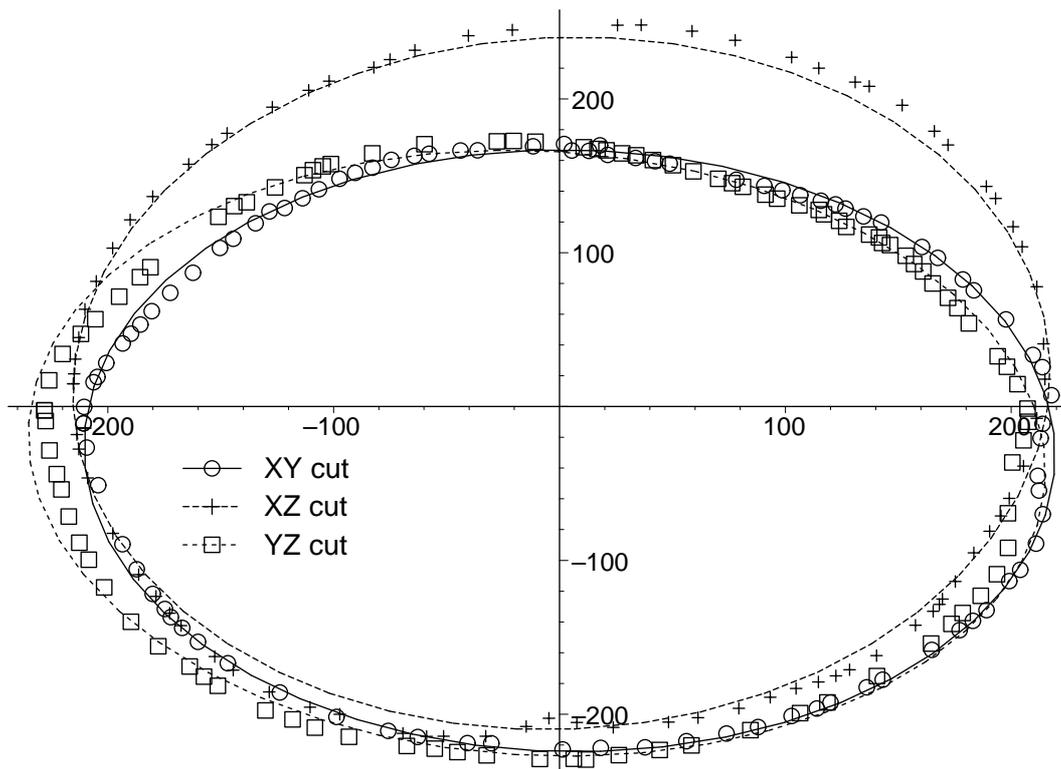
```

> ZeroMatrix(Nb,Nc),Matrix(zip((u,v)->subs(fb0=u,X=v[1],Z=v[2],JZb),Fb,XZ)),
> [ZeroMatrix(Nc,Na),ZeroMatrix(Nc,Nb),DiagonalMatrix(map(u->subs(fc0=u,JFyc),
> Fc)),Matrix(zip((u,v)->subs(fc0=u,Y=v[1],Z=v[2],JYc),Fc,YZ)),
> [ZeroMatrix(Nc,Na),ZeroMatrix(Nc,Nb),DiagonalMatrix(map(u->subs(fc0=u,JFzc),
> Fc)),Matrix(zip((u,v)->subs(fc0=u,Y=v[1],Z=v[2],JZc),Fc,YZ))]):
> R:=-Vector([zip((u,v)->subs(fa0=u,X=v[1],Y=v[2],RXa),Fa,XY)[],
> zip((u,v)->subs(fa0=u,X=v[1],Y=v[2],RYa),Fa,XY)[],zip((u,v)->subs(fb0=u,
> X=v[1],Z=v[2],RXb),Fb,XZ)[],zip((u,v)->subs(fb0=u,X=v[1],Z=v[2],RZb),
> Fb,XZ)[],zip((u,v)->subs(fc0=u,Y=v[1],Z=v[2],RYc),Fc,YZ)[],
> zip((u,v)->subs(fc0=u,Y=v[1],Z=v[2],RZc),Fc,YZ)[]):
> DV:=LeastSquares(J,R):eps:=Norm(DV,2):DV:=0.5*DV;
> Fa:=Fa+convert(DV[1..Na],list):Fb:=Fb+convert(DV[1+Na..Na+Nb],list):
> Fc:=Fc+convert(DV[1+Na+Nb..Na+Nb+Nc],list):A0:=A0+DV[Na+Nb+Nc+1];
> B0:=B0+DV[Na+Nb+Nc+2]; C0:=C0+DV[Na+Nb+Nc+3];Ax0:=Ax0+DV[Na+Nb+Nc+4];
> Ay0:=Ay0+DV[Na+Nb+Nc+5]; Bx0:=Bx0+DV[Na+Nb+Nc+6];Bz0:=Bz0+DV[Na+Nb+Nc+7];
> Cy0:=Cy0+DV[Na+Nb+Nc+8]; Cz0:=Cz0+DV[Na+Nb+Nc+9];Ra0:=Ra0+DV[Na+Nb+Nc+10];
> Rb0:=Rb0+DV[Na+Nb+Nc+11]; Rc0:=Rc0+DV[Na+Nb+Nc+12];
> Fa:=map(u->op(2,polar(cos(u)+I*sin(u))),Fa):Fb:=map(u->op(2,polar(cos(u)+
> I*sin(u))),Fb):Fc:=map(u->op(2,polar(cos(u)+I*sin(u))),Fc):
> Ra0:=op(2,polar(cos(Ra0)+I*sin(Ra0))):Rb0:=op(2,polar(cos(Rb0)+I*sin(Rb0))):
> Rc0:=op(2,polar(cos(Rc0)+I*sin(Rc0))):
> end do:

```

## 6. Results interpretation

The accuracy of the approximation of individual cross sections can be displayed, see Figure 3.



**Fig. 3:** Accuracy of the individual cross sections.

```

> Xa:=A0*cos(f); Ya:=B0*sin(f):
> Xra:=cos(Ra0)*Xa-sin(Ra0)*Ya+Ax0: Yra:=sin(Ra0)*Xa+cos(Ra0)*Ya+Ay0:
> # similarly A0,C0,f-->Xb,Zb-->Xrb,Zrb; B0,C0,f-->Yc,Zc-->Yrc,Zrc;
> P1:=plot(XY,style=point,symbol=circle,symbolsize=20,color=black):
> P2:=plot([Xra,Yra,f=0..2*Pi],color=black,thickness=2,thickness=2,linestyle=1):
> # similarly XZ,symbol=cross --> P3, Xrb,Zrb,linestyle=3 --> P4
> # similarly YZ,symbol=box --> P5, Yrc,Zrc,linestyle=4 --> P6
> display({P1,P2,P3,P4,P5,P6});

```

Correlation between vertex points and corresponding points on the individual ellipses and the average of the relative displacements can be computed.

```

> R1:=map(u->sqrt(u[1]^2+u[2]^2),XY):
> R2:=map(u->evalf(subs(f=u,sqrt(Xra^2+Yra^2))),Fa):
> DR:=zip((u,v)->abs(u-v),R1,R2): AR:=zip((u,v)->0.5*(u+v),R1,R2):
> RRa:=zip((u,v)->u/v,DR,AR): MRa:=sum(RRa[i],i=1..Na)/Na:
> CRa:=stats[describe,linearcorrelation](R1,R2):
> # similarly XZ-->R1, Fb-->R2, R1,R2-->MRb,CRb
> # similarly YZ-->R1, Fc-->R2, R1,R2-->MRc,CRc
> Correlations:=[CRa,CRb,CRc];
> Mean_relative_displacement:=[MRa,MRb,MRc];

```

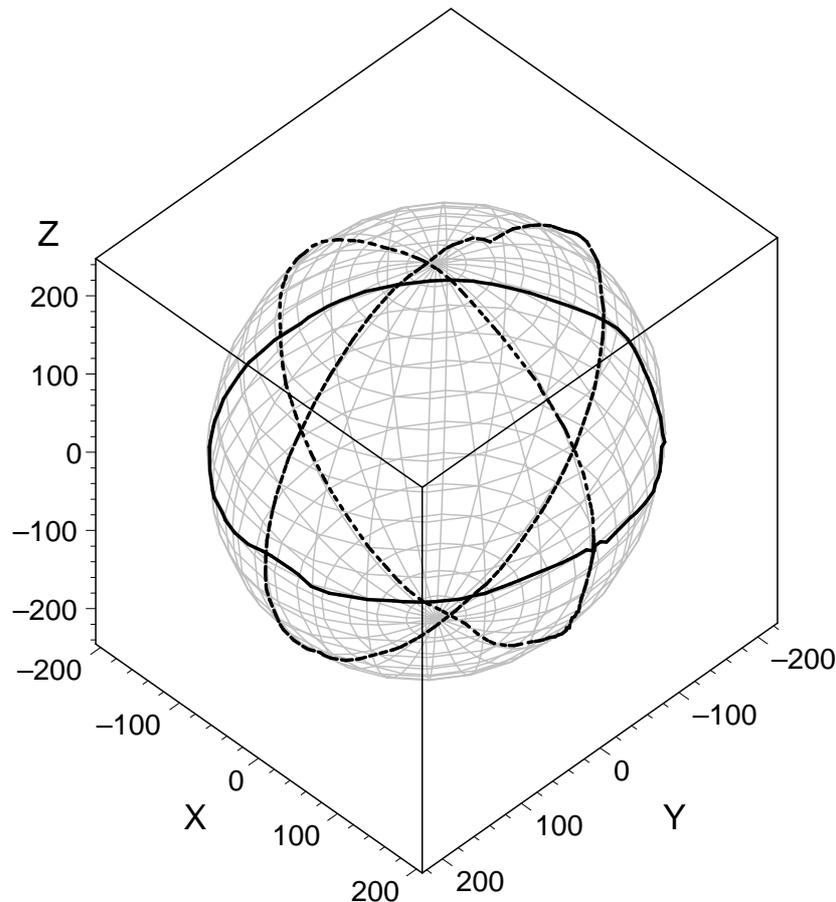
$Correlations := [0.992, 0.948, 0.084]$   
 $Mean\_relative\_displacements := [0.012, 0.021, 0.021]$

We see that mean displacements are close to 2%, the highest individual displacement is lower than 7%, and the correlation coefficient between vertices and corresponding points on the individual ellipses is higher than 90%. These results enable to declare the approximation as successful. Finally, 3D plot showing triaxial ellipsoid with projections of the cross sections can be displayed, see Figure 3.

```

> PCH:=plot3d([A0*cos(f)*cos(l),B0*cos(f)*sin(l),C0*sin(f)],
> l=-Pi..Pi,f=-Pi/2..Pi/2,style=wireframe,color=grey):
> Sa:=spacecurve(map(u->[cos(Ra0)*(u[1]-Ax0)+sin(Ra0)*(u[2]-Ay0),
>   -sin(Ra0)*(u[1]-Ax0)+cos(Ra0)*(u[2]-Ay0),0],XY),
>   color=black,thickness=3,linestyle=1):
> # similarly Bx0,Bz0,Rb0,XZ,linestyle=3 --> Sb
> # similarly Cy0,Cz0,Rc0,YZ,linestyle=4 --> Sc
> display({PCH,Sa,Sb,Sc},scaling=constrained,axes=boxed);

```



**Fig. 4:** 3D Peach approximation.

## 7. Conclusions

Regarding above stated high correlation coefficients and relative displacements, and with the reference to the results shown in Figures 3 and 4, it can be concluded that presented approach provides excellent results. It means that it can be used as a tool for the space approximation of simple three-dimensional objects, such as peaches, apricots, apples, cucumbers, etc.

## References

- [1] J. Buchar, Š. Nedomová, L. Severa: *High strain rate behaviour of peaches*. Proceedings of the 2008 SEM XI International Congress and Exposition on Experimental and Applied Mechanics, USA, Society for Experimental Mechanics, Inc., 2008, pp. 163–170. ISBN 0-912053-99-2.
- [2] P. Dvořáková, Š. Nedomová, L. Severa, J. Trnka, J. Buchar: *The impulse response method for measuring the overall firmness of fruit*. Book of Contributions of 46th International Scientific Conference “EXPERIMENTAL STRESS ANALYSIS 2008”. Ostrava, Czech Republic: VŠB – Technical University of Ostrava, 2008, pp. 43–47. ISBN 978-80-248-1774-3.
- [3] L. Severa: *Shape and strength of Red Haven peaches at the different stages of their maturity*. Acta univ. agric. et silvic. Mendel. Brun., 2008, LVI, No. 4, pp. 169–177. ISSN 1211-8516.
- [4] S. Bartoň: In: J. Diblík, (Ed.), *Shape determination of an agricultural product* Proceedings of 6. Math. workshop, Brno, FAST VUT Brno, 2007, pp. 1–12. ISBN 80-214-2741-8.
- [5] W. Gander and S. Bartoň: *Least Squares Fit with Piecewise Functions*. In: W. Gander, J. Hřebíček (Eds.), *Solving problems in Scientific Computing using Maple and Matlab*, Heidelberg, Springer, 2004, pp. 433–450. ISBN 3-540-21127-6.
- [6] W. Gander and S. Bartoň: *Metod najmensich kvadratov dlja kusocno-nepreryvnykh funkcij*. In: W. Gander, J. Hřebíček (Eds.), *Resenie zadac v naucnych vycislenijach s primeneniem Maple i Matlab*, Minsk, Vassamedia, 2005, pp. 417–442. ISBN 985-6642-06-X.