Computers in Biology and Medicine 77 (2016) 285–296

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/cbm

Analysis of cornea curvature using radial basis functions – Part II: Fitting to data-set

G.W. Griffiths [a,*], Ł. Płociniczak [b], W.E. Schiesser [c]

[a] City University, Northampton Square, London, EC1V 0HB, UK
[b] Faculty of Pure and Applied Mathematics, Wrocław University of Science and Technology, Wyb. Wyspiańskiego 27, 50–370 Wrocław, Poland
[c] Lehigh University, D118 Iacocca, 111 Research Drive, Bethlehem, PA 18015, USA

ARTICLE INFO

Article history:
Received 9 November 2015
Received in revised form
24 May 2016
Accepted 4 June 2016

Keywords:
Corneal topography
Cornea data-set
Meshless method
Radial basis function (RBF)
Radius of curvature
Partial differential equation (PDE)
Elasticity coefficient
Tension

ABSTRACT

In part I we discussed the solution of *corneal curvature* using a 2D *meshless method* based on *radial basis functions* (RBFs). In Part II we use these methods to fit a full nonlinear thin membrane model to a measured data-set in order to generate a topological mathematical description of the cornea. In addition, we show how these results can lead to estimations for corneal radius of curvature and certain physical properties of the cornea; namely, *tension* and *elasticity coefficient*. Again all calculations and graphics generation were performed using the R language programming environment.

The model describes corneal topology extremely well, and the estimated properties fall well within the expected range of values. The method is straight forward to implement and offers scope for further analysis using more detailed 3D models that include corneal thickness.

1. Introduction

This paper is a continuation of part I of the study [5], concerning a mathematical model of corneal topography. The importance of the cornea, the eye's most frontal part, cannot be overestimated since it is the place where an incident light beam is refracted directly into the interior part of the eye. Focused by the lens, the signal excites the retina from which is subsequently transmitted, via the optic nerve, into the brain. There, the image is constructed (for a technical treatment see [1]).

The cornea, as a biomechanical structure, can be characterized by both its optical and structural properties. On the one hand, it has to be almost completely transparent in order not to affect the light signal, while on the other, it should be very durable to provide a protective shield from external factors. It is a layered structure consisting of five distinguishable layers: epithelium, Bowman's layer, stroma, Descement's membrane and endothelium (from anterior to posterior). The stroma, which constitutes about 90% of the corneal thickness, is made of an arrangement of collagen fibrils and is responsible for many of its material properties. The cornea is joined with sclera (the white of the eye) with a junction called corneal limbus. A thorough exposition of the human eye anatomy can found, for example, in [8].

Corneal topography is an important factor in the process of refraction. Although, the cornea is responsible for a majority of the eye's refractive power (about 2/3), its focal point is fixed which makes the lens the main focusing organ. Any disturbance in the shape of the cornea can perturb vision quality. This is why it is important to measure and model corneal topography adequately. In earlier work [13], we have devised a medium-complexity mathematical model of corneal topography. A 1D simplified version was then studied analytically and numerically in [15,16]. Unless stated otherwise, plots were generated using case 3 data using IMQ RBFs.

The description of corneal topography data (collected for example with high-speed videokeratoscopy [23] or Scheimpflug camera [18]) requires very nontrivial mathematical modeling and various approaches have been proposed to encode (and compress) the information. We mention the adaptive modeling (for example [10]), a mixture of parametric surfaces and polynomials (see [12] and references therein), splines [6] and Zernike polynomials [3,20]. Apart from this, there are a number of different modeling approaches which are discussed, for example, in [21]. The Reader will also find a relevant comparison of models in Part I of this paper [5].

* Corresponding author.
E-mail addresses: graham@griffiths1.com (G.W. Griffiths), lukasz.plociniczak@pwr.edu.pl (Ł. Płociniczak), wes1@Lehigh.EDU (W.E. Schiesser).

http://dx.doi.org/10.1016/j.compbiomed.2016.06.008
0010-4825/

Fig. 1. Typical example of an Oculus Pentacam® instrument system. *Source*: Birmingham Optical.

In parts I and II of this paper we extend the analysis of our previous work [15] from a 1D corneal model to a full 2D model with no prior assumptions regarding radial symmetry. Part I described a solution methodology that uses a meshless method based on radial based functions (RBFs). This enabled a full topological solution to be obtained for the cornea. From this solution, radius of curvature values can be calculated over the surface. In addition, it was demonstrated that various corneal physical parameters can also be estimated from the simulation results.

We now use this methodology to fit a 2D model to a measured data-set of corneal elevations obtained from an Oculus Pentacam® device - see (Fig. 1).

2. Mathematical model

2.1. Corneal topology

Part I of this paper [5] included a *false transient* analysis and solution to cornea curvature based on the following normalized thin-plate 2D *boundary value* PDE curvature model,

$$\frac{\partial h}{\partial t} = \nabla\cdot\left(\frac{\nabla h}{\sqrt{1+|\nabla h|^2}}\right) - ah + \frac{b}{\sqrt{1+|\nabla h|^2}},$$
$$h = h(x,t), \quad t \geq 0, \quad x \in \mathbb{R}^2,$$
$$\text{BCs: } \partial\Omega = \Omega|_{r_{ab}=1} = 0, \quad r_{ab} = \left(\frac{x_1^2}{r_a^2} + \frac{x_2^2}{r_b^2}\right),$$
$$\nabla h|_{x=0} = 0, \quad \text{IC: } h(x,0) = h_0(x), \tag{1}$$

where, $a := kR^2/T$ and $b := PR/T$ (refer to Part I) and $h(x)$ represents the nondimensional height of the cornea over its surface. The maximum height $h_{\max}$, is the so-called ocular *sagitta* or *sagittal height/depth*, from geometric theory. This model is a result of a equilibrium force balance between membrane tension T, elastic restoring force proportional to the corneal deflection with the constant of proportionality k and intraocular pressure P (for derivation see [13] and discussion in Part I [5]). A typical length dimension of the cornea is here designated by R, which we choose to be the semi-major axis of the cornea.

The calculations were performed over the spatial domain Ω, assumed elliptical, with boundary $\partial\Omega$. The overall topology solution was obtained using a meshless method based on radial basis functions (RBFs). This same model will be used to provide a fit to a measured data-set of corneal elevation, with tuning achieved by varying parameters a and b. Once a and b are known, we can back calculate estimates for certain physical properties of the cornea, as detailed in part I.

2.2. Radius of curvature

The important corneal property *curvature*, we define in terms of two invariants: *Gaussian* curvature

$$K = \frac{h_{xx}h_{yy} - h_{xy}^2}{(1+h_x^2+h_y^2)^2}; \tag{2}$$

and *mean* curvature

$$H = \frac{(1+h_x^2)h_{yy} - 2h_xh_yh_{xy} + (1+h_y^2)h_{xx}}{2(1+h_x^2+h_y^2)^{3/2}}, \tag{3}$$

where we define $h = h(x, y)$ in Cartesian coordinates, and we have used subscript notation to represent partial derivatives, i.e. $h_x = \partial h/\partial x$ etc. From above, and using real units, the *principal curvatures* from differential geometry are given by

$$k_1 = H + \sqrt{H^2 - K}, \quad \text{maximum curvature},$$
$$k_2 = H - \sqrt{H^2 - K}, \quad \text{minimum curvature}, \tag{4}$$

from which it follows that $K = (k_1k_2)$ and $H = (k_1 + k_2)/2$ [2]. Curvatures k_1, k_2 and H have units of reciprocal meters, while K has units of reciprocal meters squared – see also part I.

We calculate the (Gaussian) radius of curvature (ROC) from Gaussian curvature as

$$\rho = \frac{1}{\sqrt{K}} = \frac{1}{\sqrt{k_1k_2}}. \tag{5}$$

Other ophthalmology refractive measures (in diopters) derived from k_1 and k_2 are

$$\text{MS} = (n_1 - n_0)(k_1 + k_2)/2, \quad \text{(mean power or mean sphere power)},$$
$$\text{CYL} = (n_1 - n_0)(k_1 - k_2), \quad \text{(cylinder power)},$$
$$\text{GP} = (n_1 - n_0)\sqrt{k_1k_2}, \quad \text{(Gaussian power)}, \tag{6}$$

where $n_1 = 1.3375$ is the *standard keratometric index* (SKI) and $n_0 = 1.0$ represents the *refractive index* of air.

Once the model of Eq. (1) has been tuned to the measured data by varying a and b, the above curvature values can be calculated.

3. The measured cornea data-set

3.1. Data source

The cornea data-set used to test the mathematical model and the RBF meshless solution method was obtained in January 2007 from an Oculus Pentacam® HR system with software version: 1.16r03. The Pentacam consists primarily of a *rotating mirror* and a *blue LED source*. It uses a rotating camera to derive a precision measurement elevation map of the cornea - refer to [11] for discussion on accuracy. The imaging technology is based upon the *Scheimpflug* principle[1] [19].

The data formed part of previous research [17], for which approval from the university human research ethics committee was obtained before commencement of the study. All subjects gave written informed consent to participate, and were treated in accordance with the tenets of the Declaration of Helsinki.

3.2. The data-set

The topography data-set discussed in this paper has been taken from a 25 year old healthy subject with a normal cornea. The data is

[1] Named after *Theodor Scheimpflug*, an Austrian army captain (1865–1911).

anonymized and consists of elevation values measured at locations over the *anterior* surface together with corresponding *x*–*y* coordinates for each point. The data also includes the corresponding value for intraocular pressure, as measured by a tonometer. Posterior data was made available but is not discussed here.

The raw data was provided in the form of a spreadsheet with each elevation being assigned corresponding *x*–*y* coordinates. The elevations were given in micron units, with zero at the corneal apex and increasing towards the *corneal-limbal ring* (CLR). For simulation purposes they were inverted such that the peripheral values became zero with the maximum elevation at the apex. The values were converted to millimeters by dividing by 1000. The data-set consisted of 9521 *x*–*y* coordinated elevations, uniformly scattered over the cornea anterior surface, see (Fig. 2).

Precision measurements of the corneal sagittal height and CLR were not available for this work. Also, the Pentacam data points at the edge of the analyzed area could not be reliably interpreted as delineating the CLR. However, from an examination of the Pentacam elevation data, the greatest elevation was determined to be 2.631 mm, and this value was used as a reference point for saggital height. From plotted data, we estimated the horizontal and vertical diameters to be 12.05 and 12.00 mm respectively, giving an eccentricity of 0.091 at the estimated zero datum. Model parameters were varied to obtain the final accepted model results, as described in (Section 4).

4. Results

4.1. Main solution

The overall solution is shown as a surface plot in (Fig. 3), and a summary of the results is given in (Table 1). Three solutions of the problem were obtained with differing numbers of nodes – see (Table 1). The problem converged smoothly and quickly to the final solution – see part I for more details. Plots were generated using case 3 data using IMQ RBFs.

In order to be able to compare simulated to measured elevations, it is necessary to convert model normalized values to the real measurement scale. This was achieved by noting that, from measured data, the horizontal and vertical diameters were found to be 12.05 and 12.0 mm respectively, giving a domain eccentricity of 0.091. These values corresponded to normalized simulation model values $r_a=1$ and $r_b=0.9959$, giving a scale factor of 6.025.

Unlike in part I, where we had a target central corneal height to aim for, here our target is to aim for the best overall fit of simulation elevations to measured data taken over the cornea surface. This was achieved with parameters $a=1.54$ and $b=2.15$, when the simulation normalized height at the cornea center was equal to 0.4367, equivalent to 2.631 mm.

The values chosen for a and b were determined iteratively by running a series of simulations and selecting the pair that minimized the maximum elevation error between measured and simulated cornea height h, taken over the surface. It was found that the values of a and b each affect both the maximum elevation and the radius of curvature of the cornea. Increasing the value of a has the effect of reducing the maximum elevation while at the same time increasing central curvature. On the other hand, increasing b has the opposite effect by increasing the maximum elevation and reducing overall curvature. Thus, a and b tend to have competing

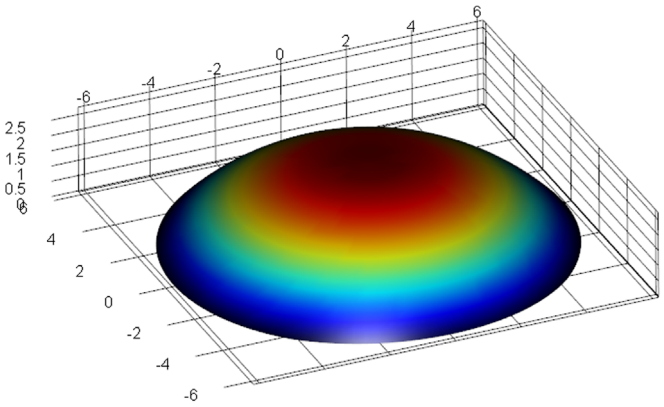

Fig. 3. 3D surface plot of cornea model solution. Dimensions are in mm.

Table 1
Results summary for simulation of Eq. (1) using the RBF meshless method with parameter values $a=1.54$ and $b=2.15$. Values are given in mm for horizontal corneal diameter (HVID), vertical cornea diameter (VVID) and radius of curvature ρ_0 for multiquadratic (MQ) and inverse multiquadratic (IMQ) RBFs, along with the corresponding shape parameters, ϵ. The results are based on the simulated central corneal elevation of 2.631 mm.

Case	HVID		VVID		ρ_0		ϵ		Total nodes
	MQ	IMQ	MQ	IMQ	MQ	IMQ	MQ	IMQ	
1	12.05	12.00	12.05	12.00	8.155	8.157	1.5	1.3	289
2	12.05	12.00	12.05	12.00	8.157	8.158	2.2	2.0	529
3	12.05	12.00	12.05	12.00	8.158	8.156	4.0	2.8	961

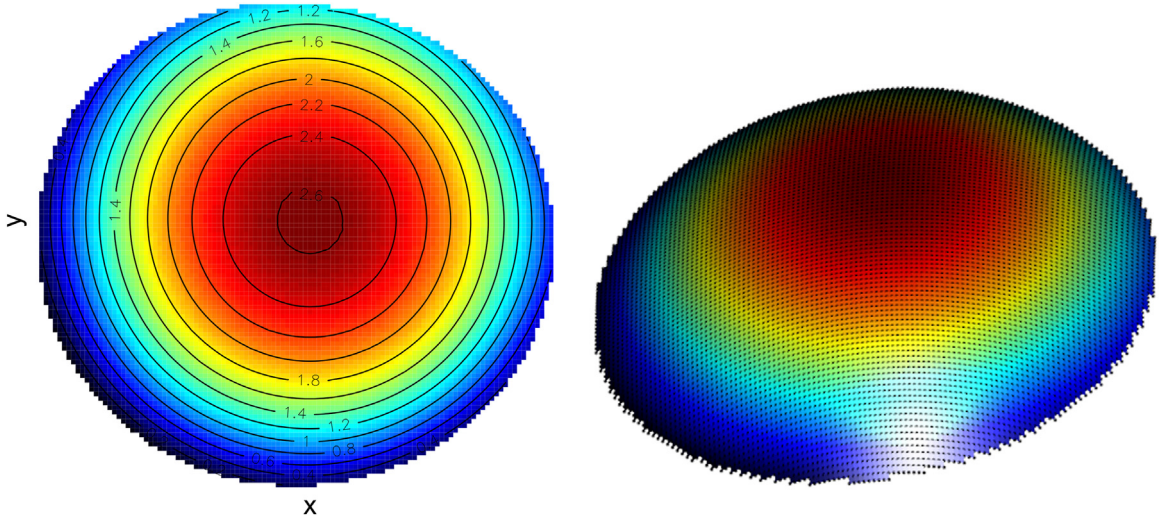

Fig. 2. *Left*: 2D surface plot of cornea data-set shown with elevation contours in mm. *Right*: 3D surface plot of cornea with data points superimposed.

effects, with b dominating. However, it was necessary to vary both a and b in order to obtain a good fit to measured data, whereby maximum curvature (min ρ_0) occurs in the central region and falls off towards the corneal periphery. Interestingly, the final values arrived at for a and b are close to those found in part I for the generic cornea. Three simulation cases were run to ensure that sufficient collocation points were used to provide adequate resolution of the corneal topography, with results shown in (Table 1).

All three cases give very similar results for corneal central height $h_{\max}$, located at (0, 0), and we take the converged value to be 2.631 mm. However, the corneal central radius of curvature $\rho_0 = \rho(0, 0)$ results differ slightly with the mean being around 8.156 mm. Both results are generally in line with published measured data and this confirms the utility of the methodology outlined in part I for estimating corneal topology.

It was found that solutions were not sensitive to the initial condition and both $h_0(x) = 0$ and $h_0(x,) = 1$ gave the same results. However, as $h_0(x) = 0$ resulted in slightly faster convergence, this value was used for all cases.

In order to compare simulated results with the measured data-set, cross-section error plots are provided in (Fig. 4). In addition, errors between model and data-set are shown as a 2D surface plot in (Fig. 5). The contours show clearly the accurate topology representation achieved by the RBF solution to model Eq. (1), particularly around the central region of the cornea. From the information available to the authors, it is not clear whether the difference at the left hand edge in Fig. 5 is due to a departure of the model from cornea measurement data, or if it represents a convex *corneo-scleral transition* region.

All calculations were performed on the same standard PC as described in part I, with similar timings.

4.2. Radius of curvature

Applying the calculations outlined in Section 2.2 with real units, the above results enable the radius of curvature to be calculated – see Figs. 6 and 7.

In order to confirm the accuracy of the radius of curvature estimate, a circle equal to the calculated apical radius of curvature, $\rho_0 = 8.156$ mm, was overlaid on top of the central area of the corneal measured data-set and the differences calculated along the x- and y-axes, see (Fig. 8). The small absolute errors indicate a good fit and provides additional confirmation of the utility of the part I calculations.

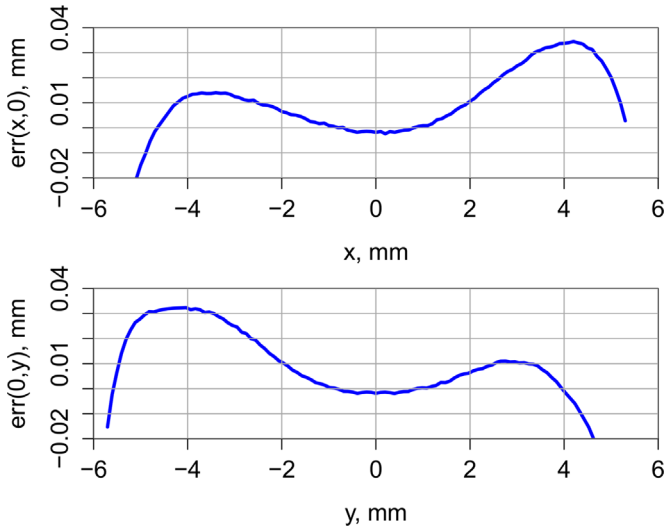

Fig. 4. Cornea anterior model solution error plots. *Top*: At cross-section, $y=0$. *Bottom*: At cross-section, $y=0$. See text.

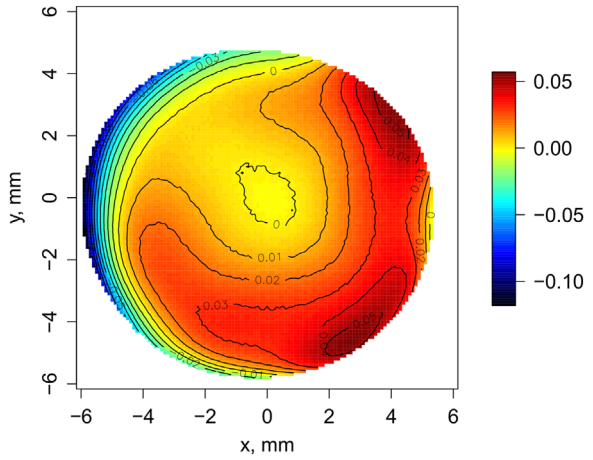

Fig. 5. Error surface color map with contours superimposed that represent the difference between model solution and the measured data-set (mm). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

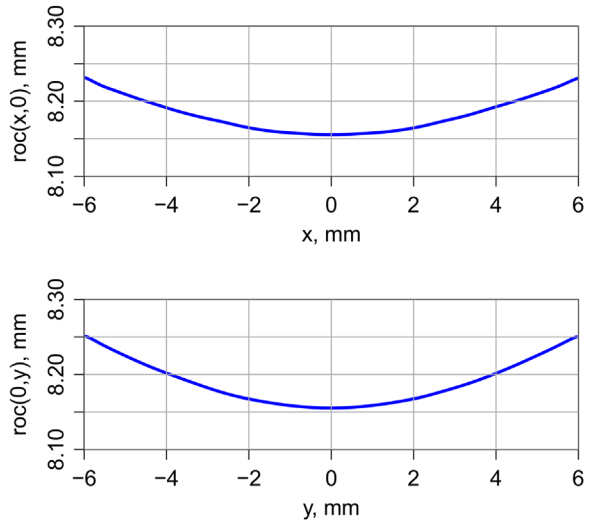

Fig. 6. *Top*: Cornea anterior radius of curvature at $y=0$. *Bottom*: Cornea anterior radius of curvature at $x=0$.

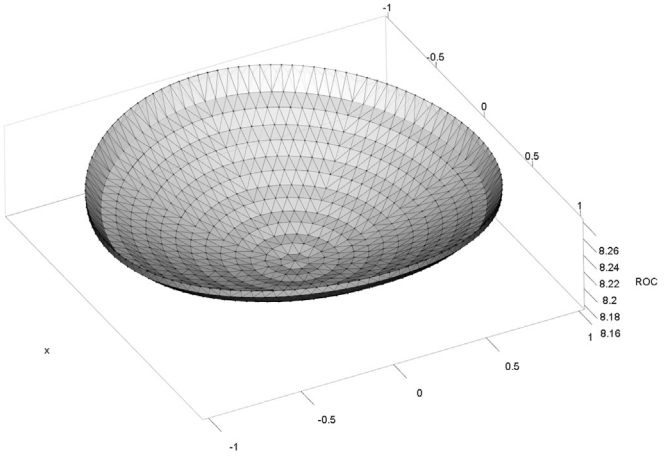

Fig. 7. Surface plot of *radius of curvature*, calculated from Eq. (5) and generated, using the R package `alphashape3d` [9].

In addition, the following values for the central cornea are also obtained:

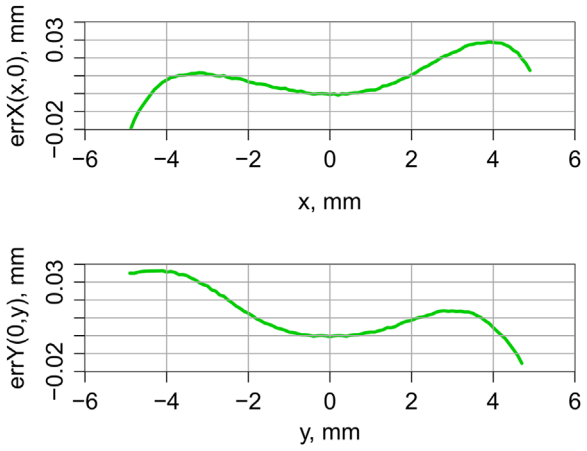

Fig. 8. Differences between the apical radius of curvature, $\rho_0 = 8.156$ mm, and the central area of the corneal measured data. *Top*: Cornea anterior differences at $y=0$. *Bottom*: Cornea anterior differences at $x=0$.

$$\begin{aligned} k_1 &= -122.1,\\ k_2 &= -123.1,\\ K &= 15026,\\ H &= -122.6,\\ \text{MS} &= 41.4,\\ \text{CYL} &= 0.4,\\ \text{GP} &= 41.4. \end{aligned} \tag{7}$$

In Eqs. (7) the curvatures k_1, k_2 and H have units of reciprocal meters, while K has units of reciprocal meters squared. These quantities relate to differential geometry. The ophthalmology refractive powers MS, CYL and GP have units of diopters – refer also to discussion in part I.

4.3. Derived physical parameters

We now use our numerical results to estimate tension T and elasticity coefficient k for the corneal central region – see discussion in part I. Using the measured value for intraocular pressure of $P = 17.8$ mm Hg (2373 N/m^2), then for a cornea having a major diameter equal to 12.05 mm, we calculate the associated tension to be

$$T = \frac{RP}{b} = \frac{0.006025 \times 2373}{2.15} = 6.65 \text{ [N/m]}. \tag{8}$$

The elasticity coefficient k, which is assumed to act as a spring constant [13], is given by

$$k = \frac{aT}{R^2} = \frac{1.54 \times 6.65}{0.\,006025^2} = 0.282 \times 10^6 \text{ [N/m}^3\text{]}. \tag{9}$$

As in part I, we emphasize that the above calculations only provide estimates based on typical corneal parameters for healthy eyes, and that further analysis and comparison with real corneal data for T and k is thus necessary.

5. Discussion

The results of the work described in Parts I and II of this paper have demonstrated that the meshless method of modeling corneal topography using radial basis functions is a viable and useful approach to analyzing the problem, yielding results that are consistent with the literature. The method can be implemented on a modest computer using the open source computer language R. The modeling approach yields:

- A good fit of the physically-based mode described in (Section 2) to a measured data-set with very small errors, by adjusting just two parameters.
- Reasonable values of ROC.
- Reasonable estimates for differential geometry values k_1, k_2, K and H.
- Reasonable estimates for opthalmology refractive powers MS, CYL, and GP.
- Estimates for T and k.

6. Further work

While the modeling work reported here constitutes a useful addition to the tools available for analyzing corneal topography, there remains much work to be done. For example, the analysis can be extended to include:

- Fitting model to additional data-sets for both healthy patients and those with abnormalities such as keratoconus. This will require that a and b are changed from fixed parameters to functions of x and y, i.e. $a(x, y)$ and $b(x, y)$. We have done some work in this area, see [14].
- Fitting model to cornea posterior surface.
- Expanding the analysis to a 3D model that includes corneal thickness.
- Consider our model as being a limiting case of the structural mechanical description of a cornea.
- Provide a mathematical treatment of non-axisymmetric perturbations in the corneal data.
- Conduct a statistical analysis of model performance in respect of corneal parameter estimates, in particular where corresponding CLR measurements are available.
- Perform further analysis of parameter estimates for T and k, and compare to measured data.

The authors hope to address some of these issues in future papers.

Conflict of interest

None declared.

Acknowledgments

The authors would like to thank Prof. D. Robert Iskander from Department of Biomedical Engineering, Wroclaw University of Technology, Poland, for access to the measured corneal data-set.

The authors would also like to thank the anonymous reviewers for their constructive suggestions and helpful comments, which have significantly improved the paper.

Appendix A. Some coding aspects

In this section we provide listings of the main R functions used in the corneal analysis.

A.1. RBF calculations

Listing 1. R code of function `mq()` used to calculate *multi-quadratic* RBF values and those of the first and second derivatives with

respect to r, x, y.

```
mq <- function(r, e, dX, dY){
  # multi-quadratic RBF function arguments
  # e = shape parameter value, scalar
  # r = radial distances, matrix
  # X = x locations, matrix
  # Y = y locations, matrix
  #
  phi     <- sqrt((e*r)^2+1)
  #
  d1rphi  <- e^2*r/phi
  d2rphi  <- e^2/phi^3
  #
  d1xphi  <- e^2*dX/phi
  d1yphi  <- e^2*dY/phi
  #
  d2xphi  <- e^2*(1+e^2*dY^2)/phi^3
  d2yphi  <- e^2*(1+e^2*dX^2)/phi^3
  return(list(phi=phi,d1rphi=d1rphi,d2rphi=d2rphi,
              d1xphi=d1xphi,d1yphi=d1yphi,
              d2xphi=d2xphi,d2yphi=d2yphi))
}
```

Listing 2. R code of function `imq()` used to calculate *inverse multi-quadratic* RBF values and those of the first and second derivatives with respect to r, x, y.

```
imq <- function(r, e, dX, dY){
  # inverse multi-quadratic RBF function arguments
  # e = shape parameter value, scalar
  # r = radial distances, matrix
  # X = x locations, matrix
  # Y = y locations, matrix
  #
  phi     <-  1/sqrt((epsilon*r)^2+1)
  #
  d1rphi  <- -epsilon^2*r/phi^3
  d2rphi  <-  epsilon^2*((2*epsilon*r)^2-1)/phi^5
  #
  d1xphi  <- -epsilon^2*dX*phi^3
  d1yphi  <- -epsilon^2*dY*phi^3
  #
  d2xphi  <-  epsilon^2*((2*dX^2-dY^2)*epsilon^2-1)*phi^5
  d2yphi  <-  epsilon^2*((2*dY^2-dX^2)*epsilon^2-1)*phi^5
  return(list(phi=phi,d1rphi=d1rphi,d2rphi=d2rphi,
              d1xphi=d1xphi,d1yphi=d1yphi,
              d2xphi=d2xphi,d2yphi=d2yphi))
}
```

A.2. Solving the PDE

Listing 3. R code of `main` program used to calculate corneal topographical solutions.

```
cat("\014")                # Clear console
rm(list = ls(all = TRUE)) # Delete workspace
#
require(compiler)
enableJIT(3)
#
library("deSolve")   # differential equation solver package
library(rgl)         # 3D interactive graphics package
library(pracma)      # advanced functions for numerical analysis
library(fields)      # provides image plot with legend
source("circCtrs.R") # node data calcs
source("nodeData.R") # node data calcs based on distmesh package
source("imq.R")      # imq RBF calcs
source("mq.R")       # mq RBF calcs
source("cornea2D_DerivList.R") # derivative calcs
#
ptm0 <- proc.time()
print(Sys.time())
#########################################################
# Note: ra, rb are major and minor axies of ellipse, where
#       ra is normalized to 1
paperNo <- 1
#########################################################
# Paper part I
if(paperNo == 1){
  aDia <- 11.75    # mm
  bDia <- 11.50    # mm
  ra <- 1          # Normalized major axis
  rb <- bDia/aDia  # Normalized minor axis
  eccentricity <- sqrt(1-(rb/ra)^2)
  eMQ  <- c(1.5,2.0,3.2)    # MQ shape parameter values
  eIMQ <- c(1.2,1.8,3.0)    # IMQ shape parameter values
  a    <- 1.48; b <- 2.104  # Best for Part I(h0=2.5)
#########################################################
# Paper part II
}else{
  aDia <- 12.05    # 12.02 # mm
  bDia <- 12.00    # mm
  ra <- 1          # Normalized major axis
  rb <- bDia/aDia  # Normalized minor axis
  eccentricity <- sqrt(1-(rb/ra)^2)
  eMQ  <- c(1.35,2.2,4.0)  # MQ shape parameter values
  eIMQ <- c(1.1,2.0,2.8)   # IMQ shape parameter values
  a    <- 1.54; b <- 2.15  # Best for Part II
}
#########################################################
# Set data for 3 cases
nDat   <- c(8,11,15)      # data point index values
caseNo <- 1               # case number
n      <- nDat[caseNo]    # number of data point
rbfType <- 0              # RBF type: 0=MQ, 1=IMQ
if(rbfType == 0){
  epsilon <- eMQ[caseNo]  # MQ epsilon value
}else{
  epsilon <- eIMQ[caseNo] # IMQ epsilon value
}
hRatio <- (aDia/2)/ra
#########################################################
# Calculate spatial domain node data
source("circDatCalc.R") # uses circCtrs() function
#
bN_index <- which(abs(xx)<1e-12 & abs(yy)<1e-12,arr.ind = TRUE) # Neuman BC
simTitle <- "Cornea_2D"
#########################################################
```

```
# 2D Cornea thin-shell Model:
# del(del_h/sqrt(1 + abs(h_x+h_y)^2 ))+a*h =
#                              b/ sqrt(1 + abs(h_x + h_y)^2 )
# Ref: W. Okrasinski, L Plociniczak (2013). Bessel function model
#      of corneal topography, App. Math. and Comp. v223, p436-443.
#
cat(sprintf("\nBoundary Nodes = %d, Inner Nodes = %d,Total Nodes = %d\n",
            Nb, Ni, N))
# Perform RBF calculations
#########################################################
XX <- outer(xx,rep(1,N),FUN="*")
YY <- outer(rep(1,N),yy,FUN="*")
dX <- t(XX)-XX; dY <- t(YY)-YY
r  <- sqrt(dX^2+dY^2)
if(rbfType == 0){
  rbfText <- "Multiquadratic (MQ) Globally Supported RBF"
  rbfDat  <- mq(r, epsilon, dX, dY)
}else if(rbfType == 1){
  rbfText <- "Inverse Multiquadratic (IMQ) Globally Supported RBF"
  rbfDat  <- imq(r, epsilon, dX, dY)
}
phi     <- rbfDat$phi; d1rphi <- rbfDat$d1rphi; d2rphi <- rbfDat$d2rphi
d1xphi  <- rbfDat$d1xphi; d1yphi <- rbfDat$d1yphi
d2xphi  <- rbfDat$d2xphi; d2yphi <- rbfDat$d2yphi
# dudxy - general applies to all RBFs
d2xyphi <- (dX*dY/r^2)*d2rphi - (dX*dY/r^3)*d1rphi # Fasheur, p443
pts <- which(abs(r)<1e-16)
d2xyphi[pts] <- 0
#
cat(sprintf("\nRBF Type = %s, epsilon = %5.2f\n",rbfText, epsilon))
#########################################################
# A <- phi
invA <- inv(phi)
#########################################################
Dx  <- d1xphi%*%invA
Dy  <- d1yphi%*%invA
Dxx <- d2xphi%*%invA
Dyy <- d2yphi%*%invA
Dxy <- d2xyphi%*%invA
#########################################################
t1  <- proc.time()-ptm0
cat(sprintf("\nTime to calculate RBFs, 1st and 2nd derivatives,
            t1 = %f\n", t1[3]))
# Boundary Conditions
Ubcr1 <- rep(0,Nb) # Dirichlet BC, fixed
Ubcr0 <- 0         # Neumann BC, fixed
# Initial condition
Uini <- rep(0,N)  # Total nodes
Uini[b_index] <- Ubcr1
#
#################################################
# Set simulation times
#################################################
tf <- 5; nout <- 51
t <- seq(from=0,to=tf,length.out=nout)
#
#################################################
# ODE integration
#################################################
ncall<<-0
#
out<-ode(method="lsodes",y=c(Uini), times=t,
         func=cornea2D_derivList,
         sparsetype = "sparseint", ynames = FALSE,
         rtol=1e-8,atol=1e-8,parms=NULL)
```

```
#
#####################################################
# Create plot array
#####################################################
# Define plot data
UU<- out[,2:(N+1)]
U <- UU[nout,]

hmax <- max(U)
cat(sprintf('\nMaximum normalized height of cornea = %f\n',hmax))
#
tCPU <- proc.time()-ptm0
cat(sprintf("Calculation time: %f\n",tCPU[3]))
#
plotOn <- 1
source("interpPlot.R")
if(paperNo == 2){
  source("corneaSimDataComparison.R")
}

source("corneaCurvature.R")
if(paperNo == 2){
  source("rocPlotOverlay.R")
}
source("cornea2D_withAlphashape3d.R")
#
tFinish <- proc.time()-ptm0
cat(sprintf("Elapsed time: %f\n",tFinish[3]))
```

Listing 4. R code of function `cornea2D_derivList()` that calculates numerical derivatives used by the `lsodes` integrator.

```
cornea2D_derivList <- function(t, u, parms) {
  ncall <<- ncall +1
#
  u[b_index] <- Ubcr1
  dhdx   <- Dx %*% u
  dhdy   <- Dy %*% u
  #print(u)
  dhdx[bN_index]<-Ubcr0 # Neuman BC
  dhdy[bN_index]<-Ubcr0 # Neuman BC
  d2hdx  <- Dx %*% dhdx[,1]
  d2hdy  <- Dy %*% dhdy[,1]
  d2hdxy <- Dx %*% dhdy[,1]
  #
  sr <- sqrt(rep(1,N)+dhdx^2 + dhdy^2)
  #
  ut <- (d2hdx+d2hdy+d2hdx*(dhdy)^2+d2hdy*(dhdx)^2 -
           2*dhdx*dhdy*d2hdxy)/sr^3 - a*u + b/sr
  ut[b_index] <- 0 # Dirichlet BC's
#
  return(list(ut))
}
```

A.3. Interpolating scattered data

Listing 5. R code of function `meshInterp()` that interpolates scattered data.

```
meshInterp2D <- function(invA,ctrs,f,epsilon,X,Y) {
  # Function to interpolate scattered data
  # invA     = inverse of RBF matrix A
  # ctrs     = vector of points used to calculate A
  # f        = vector of known values at ctrs
  # epsilon = shape parameter used to calculate A
  # X, Y     = location matrices at which f is to be interpolated
  # - see interp2D-Ex2-3options.R in folder "interpolation2D_Ex2"
  c <- invA%*%f # soln to Ac=f
  n <- dim(X)
  k <- nrow(ctrs)
  Pf<-array(0,c(n[1],n[2]))
  for( i in 1:n[1]){
    for( j in 1:n[2]){
      # Euclidean distances between est point f(i,j) and ctrs
      r<-sqrt((X[i,j]-ctrs[1:k,2])^2+(Y[j,i]-ctrs[1:k,1])^2)
      # Calculate rows of B matrix, one for each interpolated point.
      # NOTE: One could build full matrix B and obtain the solution
      # Pf = B %*% c. But for many est points, B would be very large,
      # i.e. size = n[1]*n[2] rows by k cols. Also Pf would be a
      # column matrix of length n[1]*n[2], which is not so convenient
      # for plotting purposes.
      if(rbfType == 0){        # MQ
        B <- sqrt((r*epsilon)^2 + 1)   # length=k
      }else  if(rbfType == 1){ # IMQ
        B <- 1/sqrt((r*epsilon)^2 + 1) # length=k
      }else  if(rbfType == 2){ # Wend30 RBF
        B <- pmax((1-r*epsilon),0)^2 # length=k
      }else  if(rbfType == 3){ # GMQ
        B <- (1+(r*epsilon)^2)^(5/2) # length=k
      }
      Pf[i,j] <- B %*% c
    }
  }
  return(Pf)
}
```

A.4. Plotting surfaces from scattered data

Listing 6. R code of function `interpPlot()` that generates a surface plot $h(x, y)$ and cross-section plots $h(x, 0)$ and $h(y, 0)$.

```
source("meshInterp2D.R")
ctrs <- cbind(yy,xx)
#plotOn <- 0
#
# Interpolate scattered data values for h to facilitate plotting
Np <- 21; Nt <- 121
phi   <- seq(0,pi/2,len=Np) # semi-ellisoid
# phi   <- seq(0,pi,len=Np) # ellisoid
theta <- seq(0,2*pi,len=Nt)
# grd <- meshgrid(phi,theta) # alternative to use of outer() - see below
# Phi2  <- t(grd$X); Theta2 <- t(grd$Y)
Phi   <- outer(phi,rep(1,Nt),FUN="*")
Theta <- outer(rep(1,Np),theta,FUN="*")
#ra <- 1; rb <- ra*(1-ecentricity^2)^(0.5)
#######################################################
# Convert to a mesh on an elliptical domain
XXe <- ra*sin(Phi)*cos(Theta)
YYe <- t(rb*sin(Phi)*sin(Theta))
#######################################################
# Estimated/interpolated values - elliptical domain
E <- meshInterp2D(invA, ctrs, U, epsilon, XXe, YYe)
#######################################################
if (plotOn == 1){
  jet.colors = colorRampPalette(
#   c("white","white","white","white","white","white","black")) # for eye look
#   c("white","white","white","white","lightblue","lightblue","lightblue","
    black")) # for eye look

  c("#00000F","#00007F", "blue", "#007FFF",
    "cyan", "#7FFF7F", "yellow",
    "#FF7F00", "red", "#7F0000"))
# Set palette
pal <- jet.colors(100)
#
open3d()
bg3d("white")
#
brks <- c(seq(-0.1,1,len=100))
# Set colour indices of each point for pers3d()
col.ind <- cut(E,100) #breaks=brks)#

# persp3d(XXe*hRatio,t(YYe)*hRatio,E*hRatio,color=pal[col.ind], # for real
    units
persp3d(XXe,t(YYe),E,color=pal[col.ind], # "lightblue",
        ylab = "", xlab = "", zlab = "",#back="lines",
        #ylim = c(-6,6), xlim = c(-6,6),zlim=c(0,2.7),
        box=FALSE,smooth=TRUE,axes=TRUE)
um <- c( 0.9166561, -0.3996206, -0.005590701,    0,
         0.1969353,  0.4394725,  0.876394808,    0,
        -0.3477703, -0.8044596,  0.481548250,    0,
         0.0000000,  0.0000000,  0.000000000,    1)
UM <- matrix(data=um, byrow=TRUE,nrow=4,ncol=4)
rgl.viewpoint(fov=0) # set before userMatrix
par3d(userMatrix=UM)
par3d(windowRect=c(20,100,820,900), zoom=1) # use zoom=100 for WebGL
bg3d(color=c("white","white"))
grid3d(c("x+", "y+", "z"),col="black",lty=2,lwd=1)
#axes3d((c('x', 'y', 'z')))
aspect3d(2*ra,2*rb,hmax) # place after decorate
# points3d(XXe,t(YYe),E,color ="green")
# rgl.snapshot("cornea2D_modelSurface.png", fmt="png") # print to file
```

```
}
#
# Central cross-section values for height of cornea
Ncs <- 123 # No est points (Same as data if fitting to data-set)
# Allocate y values at x=0
YYx0 <- array(seq(-rb,rb,len=Ncs),c(Ncs,1))
X0   <- array(0,c(1,Ncs)) # x=0
# Estimated h(0,y)
# Eh0Y <- meshInterp2D(invA, ctrs, U, epsilon,
#                      X0,YYx0)
####################
# Used for nodes calculated by nodeDatCalc.R (distMesh software)
# yAx_index <- which(abs(xx)<0.01,arr.ind=TRUE)
# xAx_index <- which(abs(yy)<0.01,arr.ind=TRUE)
####################
Eh0Y <- splinefun(yy[yAx_index],U[yAx_index], method = "natural")(YYx0)
#
#Eh0Y[1,c(1,Ncs)] <- 0 # ensure interpolted values are zero at end points
# Allocate x values at y=0
XXy0 <- array(seq(-ra,ra,len=Ncs),c(Ncs,1))
Y0   <- array(0,c(1,Ncs)) # y=0
# Estimated h(x,0)
# EhX0 <- meshInterp2D(invA, ctrs, U, epsilon,
#                      XXy0,Y0)
EhX0 <- splinefun(xx[xAx_index],U[xAx_index], method = "natural")(XXy0)
#EhX0[c(1,Ncs),1] <- 0 # ensure interpolted values are zero at end points
if(plotOn ==1){
  # Plot central h cross-sections
  par(mfrow=c(2,1))
  par(mar=c(5, 4, 1, 3)+0.1) # (bottom, left, top, right)
  plot(YYx0*(aDia/2),Eh0Y*(aDia/2),type="l",col="blue",asp=1,xlim=c(-6,6),ylim=
      c(0,3),
        xaxs="i",yaxs="i",xlab="y, mm",ylab="h(0,y), mm",lwd=3)#,
        #main="Central cross-section, h(0,y)")
  grid(lty=1,col="darkgray")
  plot(XXy0*(aDia/2),EhX0*(aDia/2),type="l",col="blue",asp=1,xlim=c(-6,6),ylim=
      c(0,3),
        xaxs="i",yaxs="i",xlab="x, mm",ylab="h(x,0), mm",lwd=3)#,
        #main="Central cross-section, h(x,0)")
  grid(lty=1,col="darkgray")
  par(mfrow=c(1,1))
  #
}
```

References

[1] D. Atchison, G. Smith, Optics of the Human Eye, Butterworth-Heinemann, Oxford, 2000.

[2] B.A. Barsky, Geometry for analysis of corneal shape, in: Ren-Hong Wang (Ed.), Computational Geometry: Lectures at the Morningside Center of Mathematics, American Mathematical Society, Rhode Island, USA, 2003, pp. 33–46.

[3] L.A. Carvalho, Accuracy of Zernike polynomials in characterizing optical aberrations and the corneal surface of the eye, Investig. Ophthalmol. Vis. Sci. 46 (6) (2005) 1915–1926.

[5] G.W. Griffiths, Ł. Płociniczak, W.E. Schiesser, Analysis of cornea curvature using radial basis functions – part I: methodology, Comput. Biol. Med. (2016), Submitted for publication.

[6] M.A. Halstead, B.A. Barsky, S.A. Klein, R.B. Mandell, A spline surface algorithm for reconstruction of corneal topography from a videokeratographic reflection pattern, Optom. Vis. Sci. 72 (11) 821–827.

[8] J.J. Kanski, B. Brad, Clinical Ophthalmology: A Systematic Approach, Saunders Ltd, 2015, http://www.elsevierhealth.co.uk/kanskis-clinical-ophthalmology-9780702055720.html?gclid=CJLIgaywr80CFcYcGwodznoPbw#panel1.

[9] T. Lafarge, B. Pateiro-López, A. Possolo, J. Dunkers, R implementation of a polyhedral approximation to a 3D set of points using the α-shape, J. Stat. Softw. 56 (4) (2014) 1–19, ISSN 1548–7660, Available on-line at: ⟨http://www.jstatsoft.org/v56/i04⟩(accessed July 2015).

[10] A. Martínez-Finkelshtein, D. Ramos-López, G.M. Castro-Luna, J.L. Alió, Adaptive cornea modeling from keratometric data, Investig. Ophthalmol. Vis. Sci. 52 (2011) 4963–4970.

[11] C. McAlinden, J. Khadka, K. Pesudovs, A comprehensive evaluation of the precision (repeatability and reproducibility) of the oculus Pentacam HR, Investig. Ophthalmol. Vis. Sci. 52 (10) (2011) 7731–7737.

[12] R. Navarro, L. González, J.L. Hernández, Optics of the average normal cornea from general and canonical representations of its surface topography, JOSA A 23 (2) (2006) 219–232.

[13] W. Okrasiński, Ł. Płociniczak, Bessel function model of corneal topography, Appl. Math. Comput. 223 (2013) 436–443.

[14] Ł. Płociniczak, W. Okrasiński, Regularization of an ill-posed problem in corneal topography, Inverse Probl. Sci. Eng. 21 (6) (2013) 1090–1097.

[15] Ł. Płociniczak, G.W. Griffiths, W.E. Schiesser, ODE/PDE analysis of corneal curvature, Comput. Biol. Med. 53 (2014) 30–41.

[16] Ł. Płociniczak, W. Okrasński, Nonlinear parameter identification in a corneal geometry model, Inverse Probl. Sci. Eng. 23 (3) (2015) 443–444.

[17] S.A. Read, M.J. Collins, D.R. Iskander, Diurnal variation of axial length, intraocular pressure, and anterior eye biometrics, Investig. Ophthalmol. Vis. Sci. 49 (7) (2008) 2911–2918.

[18] S.A. Read, M.J. Collins, D.R. Iskander, B.A. Davis, Corneal topography with Scheimpflug imaging and videokeratography: comparative study of normal eyes, J. Cataract Refract. Surg. 35 (6) (2009) 1072–1081.

[19] T. Scheimpflug, Der photoperspektrograph und seine anwendung, Photogr. Korr 43 (1906) 516–531.

[20] J. Schwiegerling, J.E. Greivenkampand, J.M. Miller, Representation of videokeratoscopic height data with Zernike polynomials, JOSA A 12 (10) (1995) 2105–2113.

[21] S. Talu, M. Talu, An overview on mathematical models of human corneal surface, in: International Conference on Advancements of Medicine and Health Care through Technology, Springer, Berlin, Heidelberg, 2009, pp. 291–294.

[23] M. Zhu, M.J. Collins, D.R. Iskander, Dynamics of ocular surface topography, Eye 21 (5) (2007) 624–632.