NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

School of Science

Information Technologies in Medicine and Biology

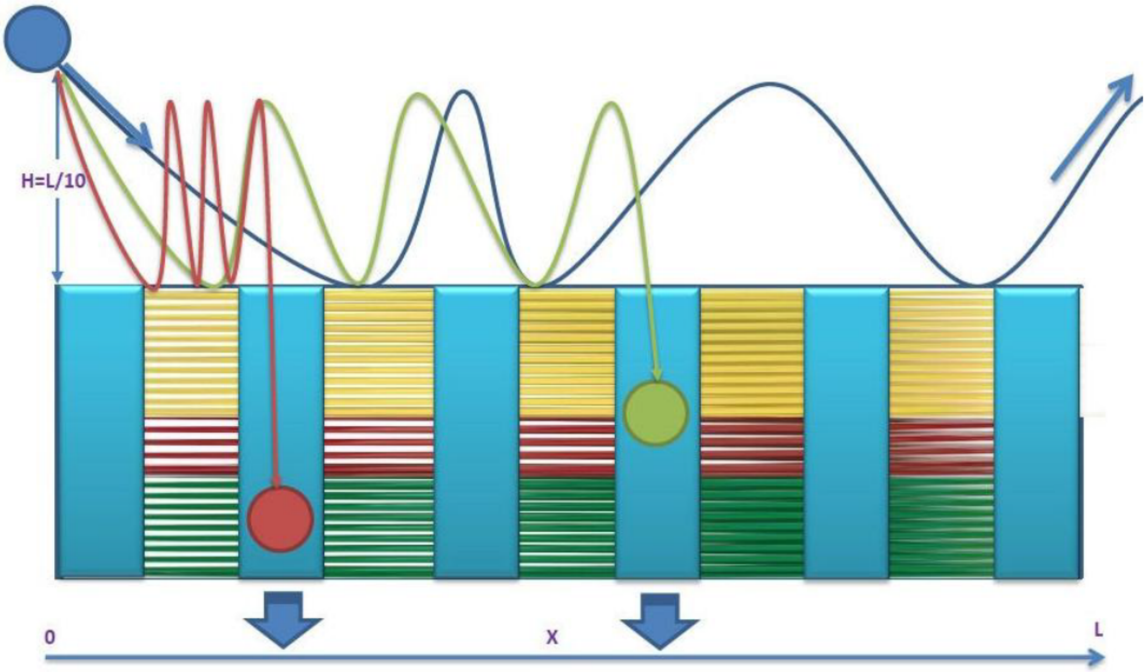
Direction: ***Bioinformatics***

Simulation Methods in Medicine and Biology

Postgraduate Student: *Begetis Nikolaos* Professor: *Spyrou George*

Date: *04/01/2017*

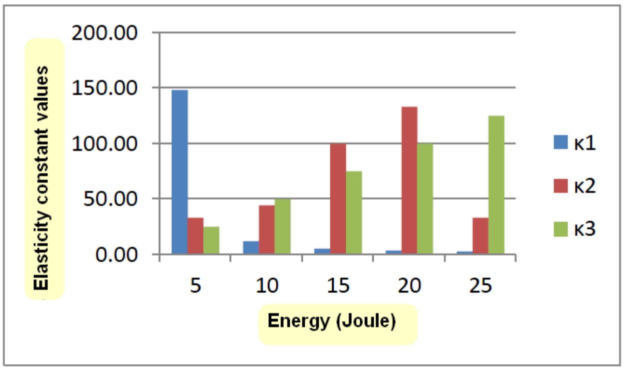
Assignment 2



E: Sphere energy, that follows a range spectrum of Energies that has been measured and the measurements are show on the accompanying table(askisi_2_supplement.xlsx).

κ: Spring constant synthesized from the constant values (k1,k2,k3) of three springs in a column. The individual spring constants are depended from the spheres' energies, and their relation has been measured as follows in the table and chart of the next page.

E	κ1	κ2	κ3
5	148.41	33.33	25.00
10	12.18	44.44	50.00
15	5.29	100.00	75.00
20	3.49	133.33	100.00
25	2.72	33.33	125.00



There exist three different ways for the spheres rebound, each one with a probability to happen:

Rebound A1	Rebound A2	Rebound A3
With probability κ1	With probability κ2	With probability κ3
When happens, the sphere's energy is preserved. (E'=E).	When happens, the sphere's energy is reduced to the half (E'=E/2).	When happens, the sphere's energy is reduced by 1/10 of the previous one (E'=E/10).

L: Length of elastic layer, L=90

λ: Route between two collisions (sphere-string) with PDF:

$$\lambda \sim e^{-kEx}$$

We throw 10^6 spheres with angles ϑ in [0^o,45^o] following uniform PDF.

At the elastic layer there exist holes every L/10 steps (the center of the hole).

Every hole's radius is R_{hole}=1.5

Every sphere's radius is R_{sphere}=0.5

Suppose that a sphere passes through the hole when the sphere's center hits in a distance from the hole's center less than R_{hole}-0.8R_{sphere}(20% tolerance).

2.1.-2.5.

We are assigned to find the number of spheres that reach the right end of the elastic layer, and the range spectrum of these, as well as the number of spheres that are falling into holes of the elastic layer, and their range spectrum, and the hole that the most of them are fallen into.

To do this first we have to find the cumulative distribution function of the angles ϑ.

This task was also asked in assignment 1, so as we had found once again we have:

Angles (ϑ)

Using the equation of uniform PDFs we have:

$$\mathbf{f}(\boldsymbol{\theta}) = \frac{1}{\beta - \alpha} = \frac{1}{\frac{\pi}{4} - 0} = \frac{\mathbf{4}}{\boldsymbol{\pi}}$$

So the cumulative distribution function of the angles ϑ using the inverse method for the p(θ) function, in order to produce 10^6 MC number, is*:*

$$F(\theta 1) = \int_0^{\theta 1} p(\theta) d\theta = \int_0^{\theta 1} \frac{4}{\pi} d\theta = \frac{4}{\pi}\int_0^{\theta 1} d\theta = \frac{4}{\pi}(\theta 1 - 0) = \frac{\mathbf{4\theta 1}}{\boldsymbol{\pi}}$$

Suppose $\frac{\mathbf{4\theta 1}}{\boldsymbol{\pi}} = \boldsymbol{R_{theta}} \rightarrow \boldsymbol{\theta 1} = \frac{\boldsymbol{\pi R}_{\mathbf{theta}}}{\mathbf{4}}, \boldsymbol{\theta} \in [\mathbf{0}, \frac{\boldsymbol{\pi}}{\mathbf{4}}]$

Route (λ)

Using the equation of inverse square power PDFs we have:

$$\boldsymbol{\lambda(x)} = \alpha e^{-kEx}, x \in [0, \infty)$$

$$\int_0^{\infty} \alpha e^{-kEx} dx = 1 \rightarrow -\frac{1}{kE}\alpha \int_0^{\infty} \left(e^{-kEx}\right)' dx = 1 \rightarrow -\frac{1}{kE}\alpha(0-1) = 1 \rightarrow \quad \boldsymbol{a = kE}$$

So the CDF of route λ(x) PDF is:

$$\Lambda(x) = \int_0^{x} kEe^{-kEx} dx = 1 \rightarrow -\int_0^{x} \left(e^{-kEx}\right)' dx = 1 \rightarrow -\left(e^{-xkE} - 1\right) = 1 \rightarrow \boldsymbol{\Lambda(x)} = \boldsymbol{e^{-xkE} + 1}, \quad \boldsymbol{x} \in [\mathbf{0}, \infty)$$

Using the inverse method, in order to produce 10^6 MC number, suppose

$\mathbf{e^{-x1kE} + 1} = \boldsymbol{R_{route}} \rightarrow \boldsymbol{x1} = \frac{\boldsymbol{ln(1 - R_{route})}}{\boldsymbol{-kE}}$

Energy (E)

In contrary to assignment 1, energy doesn't follow the inverse square PDF. So we have to extract the PDF that the energy follows from the supplementary table given in the xlsx file. This will be done using the rejection method.

Suppose I_{max} the highest value of Energy Intensity shown in the table. We divide all values with I_{max} in order to normalize all energy values in the range [0,1] *(function I)*. Then create random energy values using the uniform PDF in [5,25] and compute the normalized value,

$I(R_e)$. Every time check if each random energy value and give the respective energy intensity from the PDF created with linear interpolation, even if this value does not exist in the table.

Then, again create random energy values in [0,1] and if compared to the previous $I(R_e)$ the second number is smaller; then keep it, else reject it and repeat the creation of random energy values in [5,25] as written above. So, as we told in class, the first random numbers create the PDF and the second random numbers take a value in the range of values that are acceptable by the instance of the PDF value of the 1st number.

Now that we are ready, in order to find all the above questions we wrote the code in the *R* (ass2.r) that follows:

```
Nspheres = 10

# run computations 6 times by multiplying Nsamples 10 times each iteration
for(i in 1:7){

	collisions = 0						# counts the collisions of each sphere
	theta_t = matrix(0,Nspheres,1)			# Nspheres' angles
	holes_t = matrix(0,1,10)
	L=90
	H=L/10
	distance = 0			# distance computed after every sphere[i] collision
	spheresExited = 0
	spheresInHoles = 0
	spheresInSpecificHole = 0
	sphereStillInElasticLayer = TRUE


# XLS Filereading

	filename = system.file("askisi_2_supplement.xlsx", package = "XLConnect")
	sheet = 1

	fileE = readWorksheetFromFile(filename, sheet, startCol = 5, startRow = 4, endCol = 5, endRow = 8)			# E range = 'E4:E8'
	fileK1 = readWorksheetFromFile(filename, sheet, startCol = 6, startRow = 4, endCol = 6, endRow = 8)			# k1 range = 'F4:F8'
	fileK2 = readWorksheetFromFile(filename, sheet, startCol = 7, startRow = 4, endCol = 7, endRow = 8)			# k2 range = 'G4:G8'
	fileK3 = readWorksheetFromFile(filename, sheet, startCol = 8, startRow = 4, endCol = 8, endRow = 8)			# k3 range = 'H4:H8'
	fileEnergies = readWorksheetFromFile(filename, sheet, startCol = 1, startRow = 4, endCol = 1, endRow = 104)	# Energy range = 'A4:A104'
	fileIntensities = readWorksheetFromFile(filename, sheet, startCol = 2, startRow = 4, endCol = 2, endRow = 104)	# Intensity range = 'B4:B104'

	Rtheta = runif(Nspheres, min=0, max=1)# producing Nspheres of random samples
```

```
Rtheta

	for (i in 1:Nspheres) {
		theta_t[i] = 45*Rtheta[i]				#  creating  Nspheres'  initial
angles
	}

	# creating Nspheres' energies using Rejection Method (energies)
	Imax = max(fileIntensities)
	fileIntensitiesNormalized = fileIntensities/Imax

	itrE = 0
	re_ok_flag = FALSE
	i = 1

	# linear interpolation
	while(itrE < Nspheres){
		itrPosition = 1
		Re_test[i] = runif(1, min=0, max=1)
		Re[i] = 5 + 20*Re_test[i]				#				see
ass2_nmpegetis_piv0111.pdf

		while(re_ok_flag == FALSE){
			if(Re[i] >= fileEnergies[itrPosition]){
				itrPosition = itrPosition+1
			}
			else{
				itrPosition = itrPosition-1
				re_ok_flag = TRUE
			}
		}

		downLim = fileEnergies[itrPosition]
		downLimDistance = Re[i] - downLim
		wB = downLimDistance * 5
		wA = 1-wB
		# linear interpolation equation
		gfunc[i]		=		wA*(fileIntensitiesNormalized[itrPosition])		+
wB*(fileIntensitiesNormalized[itrPosition+1])

		Re_test2[i] = runif(1, min=0, max=1)
	# rejection method
		if(Re_test2[i] < gfunc[i]){
			energies = Re[i]
			itrE = itrE + 1
		}
		i = i + 1
		re_ok_flag = TRUE
	}
```

```
		for (s in 1:Nspheres){			# spheres routes
			print(s)
			d = tand(theta_t[s])*H	# d: distance from the first collision
			distance = d

			sphereStillInElasticLayer = TRUE
			collisions = 0

			if(distance >= 0 && distance < 2.6){			# In case that the elastic
layer begins with hole check if it falls in
				sphereStillInElasticLayer = FALSE
				holes_t[s] = holes_t[s] + 1
				spheresInSpecificHole = spheresInSpecificHole + 1
			}

			while(sphereStillInElasticLayer){
				collisions = collisions + 1
		##############exw meinei edw!!!
				shpereEnergies[s,collisions] = energies[s]

		#	[k] = grammikh_parembolh_k_elathriou(energies[s], fileE, fileK1, fileK2, fileK3)

				table_read = TRUE
				k = matrix(0,1,3)

				lineCntr = 1

				E = value_E

				while(table_read == TRUE){
					if(E >= fileE(lineCntr)){
						lineCntr = lineCntr + 1
					}
					else{
						lineCntr = lineCntr - 1
						table_read = FALSE
					} #
				} #

				# if energy is < 5J, then it will randomly take a value in [5,6]
				if(lineCntr == 0){
					lineCntr = lineCntr + 1
					E = fileE(lineCntr) + runif(1, min=0, max=1)
				} #

				downLim = fileE(lineCntr)
				downLimDistance = E - downLim
				wB = downLimDistance * 0.2
				wA = 1 - wB

				k(:,1) = wA * fileK1[lineCntr] + wB * fileK1[lineCntr + 1]
```

```
				k(:,2) = wA * fileK2[lineCntr] + wB * fileK2[lineCntr + 1]
				k(:,3) = wA * fileK3[lineCntr] + wB * fileK3[lineCntr + 1]

				k_total = sum(k)
				a1prob = k[1)/k_total
				a2prob = k[2]/k_total
				a3prob = k[3]/k_total

				Rj = runif(1, min=0, max=1)			# jumbing

				if(Rj > a1prob && (Rj <= a1prob + a2prob)){
					energies[s] = energies[s]/2
				}
				elseif((Rj > a1prob + a2prob) && Rj <= 1){
					energies[s] = energies[s]/10
				} #

				Rx = runif(1, min=0, max=1)		# random sampling Rx

				x = -(1/(k_total*energies[s]))*log(1-Rx)	# shown in report

				distance = distance + x

				if(distance > L){		# outside the elatic layer
					sphereStillInElasticLayer = FALSE
					spheresExited = spheresExited + 1
					spheresExitedOutside(spheresExited,1) = s
					break				# go to next sphere
				}
				else{							# inside elastic layer
					for (z in 1:9){				#  check  if  sphere  falls  to
layer's holes
						if((distance > z*9 + 0.4) && (distance < z*9 + 2.6)){
							sphereStillInElasticLayer = FALSE
							holes_t(:,z+1) = holes_t(:,z+1) + 1
							spheresInHoles = spheresInHoles + 1
							spheresFallenInHoles[spheresInHoles,1] = s
							break
						} #
					} #
				} #
			} #
		} #

		if(spheresExited != 0){ # if at least one sphere exited the elastic layer
			d=1
			count_zero = matrix(0,size(spheresExitedOutside,1),1)

			for (  m in 1 : size(spheresExitedOutside,1) ) {
				spheresEnergiesOutside(m,:)						=
shpereEnergies(spheresExitedOutside(m),:)
```

```
                    while (d <= size(spheresEnergiesOutside(m,:),2)){
                        if(spheresEnergiesOutside(m,d) == 0){
                            count_zero(m,1) = count_zero(m,1) + 1
                            d = d + 1
                        }
                        else{
                            d = d + 1
                        } #
                    } #

                    spherePosition = size(spheresEnergiesOutside,2) - count_zero(m,1)
                    count_zero(m,1)
                    finalEnergyValue(m,1) = spheresEnergiesOutside(m,spherePosition)
                    d=1
                } #

                # Energy Spectrum of spheres that get outside the layer
                figure
                hist(finalEnergyValue) title('Energy Spectrum of spheres that get outside the
layer')
                xlabel('Energy') ylabel('Frequency')

            } #

            if(spheresInHoles != 0){ # if at least one sphere fell into a hole
                l=1
                count_zeros = matrix(0,size(spheresFallenInHoles,1),1)

                for (t in 1:size(spheresFallenInHoles,1)) {
                    spheresEnergiesInsideHoles(t,:)                                    =
shpereEnergies(spheresFallenInHoles(t),:)

                    while (l <= size(spheresEnergiesInsideHoles(t,:),2)){
                        if(spheresEnergiesInsideHoles(t,l) == 0){
                            count_matrix(0,t,1) = count_matrix(0,t,1) + 1
                            l = l+1
                        }
                        else{
                            l = l+1
                        } #
                    } #

                    lastPositionOfFinalEnergy   =   size(spheresEnergiesInsideHoles,2)   -
count_matrix(0,t,1)
                    count_matrix(0,t,1)
                    finalEnergyOfFallenSpheres(t,1)                                    =
spheresEnergiesInsideHoles(t,lastPositionOfFinalEnergy)
                    l=1
                } #
```

```
                # Energy Spectrum of spheres that fell into holes
                figure
                hist(finalEnergyOfFallenSpheres) title('Energy Spectrum of spheres that fell
into holes')
                xlabel('Energy') ylabel('Frequency')

        } #

        fprintf('\n\nThe number of spheres that got outside the layer: \n\n', spheresExited)
        fprintf('The number of spheres that fell into holes: #_f\n\n', spheresInHoles +
spheresInSpecificHole)
        fprintf('Into hole no1: #d\nInto hole no2: #d\nInto hole no3: #d\nInto hole no4:
#d\nInto hole no5: #d\nInto hole no6: #d\nInto hole no7: #d\nInto hole no8: #d\nInto hole
no9: #d\nInto hole no10: #d\n\n',holes_t)

        [maxValue, maxSpherePosition] = max(holes_t)
        fprintf('H pio "eunohmenh" eksodos pros ta katw einai h: #_fh sthn opoia epesan #_f
apo tis sunolika #_f sfaires_\n\n', maxSpherePosition, maxValue, Nspheres)

        #fprintf('Sthn trupa 1h  epesan: #d\nSthn trupa 2h  epesan: #d\nSthn trupa 3h
epesan: #d\nSthn trupa 4h  epesan: #d\nSthn trupa 5h  epesan: #d\nSthn trupa 6h  epesan:
#d\nSthn trupa 7h  epesan: #d\nSthn trupa 8h  epesan: #d\nSthn trupa 9h  epesan:
#d\nSthn trupa 10h epesan: #d\n \n\n',holes_t)
}
```

After running the above *R* code in R console the printed results for all the questions asked were: