Introduction

Every four years the world's greatest athletes gather together at a host country to compete in a variety of sports to decide who are the three best performers to be awarded a medal for their representing country. A common question that has been asked about the Olympics is if there is a home-country advantage, where it is believed that the rate of medals per participant increases during the host year. The provided data is the number of participants and the number of medals they won for various host countries the Olympics they hosted and the previous Olympics they participated in. The data is for the Summer Olympics from 1952 to 2021. This data can be described with Y_{i1} being the number of medals won by the host country during Olympics *i* and Y_{i0} being the number of participants by the host country during Olympics. Similarly, N_{i1} and N_{i0} are the number of participants by the host country during Olympics *i* and the same country in the previous Olympics respectively. Using this data, this summary will detail the Bayesian analysis to determine if there is statistical backing for the claim that hosting the Olympics when they host, and if there is evidence that specific countries have more of a home-country advantage over others.

Aggregate Analysis

Beginning with the claim that hosting the Olympics leads to a greater medal rate, the data across all years is aggregated, i.e. $Y_1 = \sum_{i=1}^{18} Y_{i1} = 1016$, $Y_0 = \sum_{i=1}^{18} Y_{i0} = 682$, $N_1 = \sum_{i=1}^{18} N_{i1} = 7979$, and $N_0 = \sum_{i=1}^{18} N_{i0} = 4715$ being the total number of medals won by the host country, the total number of medals won by the same country during the previous Olympics, the total number of participants by the host country, and the total number of participants by the same country during the previous Olympics. To obtain posterior distributions for λ_1 and λ_0 , i.e. the expected number of medals per participate by the home country during the host year and during the previous Olympics respectively, the first step is to define what our likelihood and prior distributions are. The data can be reasonably modelled with a Poisson distribution, i.e. $P(Y|\lambda) \sim Poisson(N * \lambda)$, since the number of medals that can be greater than 1, and each medal won is independent of each other. A conjugate prior for a Poisson rate is the gamma distribution, $P(\lambda) \sim Gamma(a, b)$, with parameters a and b both equaling 0.10 so that it is uninformative. As stated previously, λ could theoretically be any positive real number and thus the gamma distribution fulfills this support. With this assumed likelihood and prior, the posterior distribution will follow a gamma distribution with parameters A = a + Y and B = b + N. Thus, the posterior distribution is $P(\lambda|Y) \sim Gamma(A, B)$.

The main assumptions of this analysis are (1) that there is no difference between the participants in the different countries so that the participants can be aggregated together, (2) the increased participation of the host country is adding participants of equal caliber as those that would be normally competing during a non-host year so that they each have equal probability of winning a medal, and (3) each medal won is independent of each other. Assumption (1) is believed to be valid since all athletes receive similar nutrition and training so that they all have equal probability of winning a medal, although this is debatable when comparing the level of funding athletes receive from country to country. Assumption (2) is also believed to be valid since all athletes and thus should all have equal probability of winning a medal, but it has been reported in the 538 reference that the qualifications are lowered in the host country so that more athletes can participate. Just because the qualifications are lowered, does not necessarily change the equal probability that everyone has to winning a medal. Assumption (3) is also believed to be valid, taking swimming as a working example, since winning the gold medal in free style does not give you any advantage in your competition in the butterfly stroke.

With the posterior defined to be $P(\lambda|Y) \sim Gamma(a + Y, b + N)$ and the assumptions of the model discussed above, the posterior distributions and their respective summaries for both the host and non-host medal rates for the aggregated data are shown in Figure 1. Comparing the two posterior distributions, one can see that the majority of the non-host posterior distribution, i.e. red curve, is above the host posterior distribution, i.e. blue curve. The spread of the host posterior is less than the spread of the non-host posterior due to more data being available to construct the host distribution. Looking at the 2.5% quantile for the non-host distribution, there is overlap of the distributions.

Figure 1: Posterior distributions and summary of host and non-host medal rate for the aggregated data.



	λ_0	λ1
Mean	0.14466	0.12735
Std	0.00003	0.00002
Quantile (2.5%)	0.13401	0.11963
Quantile (97.5%)	0.15572	0.13529

Hypothesis Test

With the posterior distributions from the previous section, a hypothesis test can be conducted to determine the probability that the host medal rate is greater than the non-host medal rate given the data, i.e. $P(\lambda_1 > \lambda_0 | Y_1, Y_2)$, to see if there is a home-country advantage. The null hypothesis, H_0 , is if $P(\lambda_1 > \lambda_0 | Y_1, Y_2) \ge$ 0.95 then we can conclude that the home-country advantage does exist. The alternative hypothesis, H_1 , is if $P(\lambda_1 > \lambda_0 | Y_1, Y_2) < 0.95$ then we can conclude that the home-country advantage does not exist. To obtain an estimate of $P(\lambda_1 > \lambda_0 | Y_1, Y_2)$, Monte Carlo (MC) sampling is employed with 100,000 samples from both posterior distributions to determine how much area in the λ_1 distribution, i.e. blue distribution, is greater than the λ_0 distribution, i.e. red distribution. From these simulations it was found that 0.506% of the λ_1 distribution is greater than the λ_0 distribution, so that one can say that there is a 0.506% chance that the true value of λ_1 is greater than the true value for λ_0 . Thus, one can be confident from these results that the home-country advantage does not contribute to a significantly greater medal rate per participant.

To determine if the above conclusion is sensitive to the selected prior of Gamma(a=0.1,b=0.1), $P(\lambda_1 > \lambda_0 | Y_1, Y_2)$ and summarizing statistics for $P(\lambda_0 | Y_0)$ and $P(\lambda_1 | Y_1)$ using different priors is shown in Table 1. Note that the summarizing statistics for $P(\lambda_0 | Y_0)$ and $P(\lambda_1 | Y_1)$ are separated with "/". As can be seen from these results, they are found to not change significantly with the chosen prior. So that we can conclude that the results are somewhat sensitive to the prior but not to a degree that would significantly impact our conclusions since the amount of data we have overpowers the prior.

Table 1: Probability that the host medal rate is greater than the non-host medal rate, i.e. $P(\lambda_1 > \lambda_0 | Y_1, Y_2)$, and summary of non-host / host posterior distributions, i.e. $P(\lambda_0 | Y_0) / P(\lambda_1 | Y_1)$, for different priors

a	b	$P(\lambda_1 > \lambda_0 Y_1, Y_2)$	Mean	Quantile (2.5%)	Quantile (97.5%)
0.10	0.10	0.00506	0.145 / 0.127	0.134 / 0.120	0.156 / 0.135
1	1	0.00491	0.145 / 0.127	0.134 / 0.120	0.156 / 0.135
2	2	0.00475	0.145 / 0.127	0.134 / 0.120	0.156 / 0.135
5	5	0.00465	0.146 / 0.128	0.135 / 0.120	0.157 / 0.136

Prediction

Knowing that France had $N_{F0} = 398$ participants and won $Y_{F0} = 33$ medals in the 2021 summer Olympics, the posterior predictive distribution (PPD) can be derived to estimate the number of medals France will win in the 2024 Olympics when they host and quantify the associated uncertainty. In order to derive the PPD, the number of participants N_{F1} must first be predicted. This is accomplished by utilizing linear regression with the provided data to predict N_1 as a function of N_0 . This fit is shown in blue in the left graph of Figure 2. The 95% prediction intervals are also shown in green and are used to include the uncertainty of N_{F1} in the PPD. With this information, it is determined that France should have about 561 ± 114 participants in 2024. Also, in Figure 2 is the linear regression fit of the provided data to predict Y_1 as a function of Y_0 and shows that France should win about 50 medals in 2024.

With an estimate of N_{F1} , the next step is to define the posterior $P(\lambda|Y)$ and the likelihood $f(Y^*|\lambda)$ to be used in a MC sampling procedure to estimate the PPD $f^*(Y^*|Y)$. The posterior is a gamma distribution, $P(\lambda|Y) \sim Gamma(0.10 + Y_{F0}, 0.10 + N_{F0})$, for the reasons explained in the aggregate study. In this analysis, the French participants and medals won in the previous Olympics are being used instead of the aggregate values calculated previously, i.e. N_1 and Y_1 , since (1) the previous French performance is assumed to be a better indicator of future performance and (2) the aggregate analysis and the country specific analysis conducted in the next section showed that the home-country advantage does not contribute to a significantly greater medal rate per participant. The likelihood is given with a Poisson distribution, $P(Y^*|\lambda) \sim Poisson(N_{F1} * \lambda)$, for the same reasons as discussed during the aggregate analysis, where N_{F1} follows a normal distribution, $N_{F1} \sim Normal(561,52)$, from the linear regression analysis. With 100,000 MC samples, the PPD in red assuming N_{F1} to have uncertainty and the PPD in blue assuming N_{F1} to be equal to 561 is shown in Figure 3. From these results we see that the PPD is not very sensitive to the N_{F1} estimate. Including N_{F1} uncertainty, there is a 95% probability that France will win between 26-72 medals in 2024 with a mean medal count of 46.64, which is close to the predicted 50 medals from linear regression.









Country-Specific Analysis

For the country specific analysis, the medal rate ratio $r_i = \lambda_{i1}/\lambda_{i0}$ is compared to determine if the homecountry advantage differs by country. Similarly, to the aggregate analysis, the posterior distributions of λ_{i1} and λ_{i0} are defined to be $P(\lambda_{i1}|Y_{i1}) \sim Gamma(a + Y_{i1}, b + N_{i1})$ and $P(\lambda_{i0}|Y_{i0}) \sim Gamma(a + Y_{i0}, b + N_{i0})$. To obtain these posterior distributions, the likelihood is a Poisson distribution, i.e. $P(Y_i|\lambda_i) \sim Poisson(N_i * \lambda_i)$, while the conjugate prior is the gamma distribution, $P(\lambda_i) \sim Gamma(a, b)$, with parameters a = b = 0.10 so that it is uninformative. The likelihood and prior were chosen for the same reasons outlined previously in the aggregate analysis. With 100,000 MC samples, the posterior distributions of the medal rate ratio for each country are shown in Figure 4 and are constructed by sampling from both $P(\lambda_{i1}|Y_{i1})$ and $P(\lambda_{i0}|Y_{i0})$ to obtain a sample from the r_i posterior. From the curves, a medal rate ratio of 1 signifies there is not a home-country advantage, less than 1 signifies the athletes performed worse in their home country, and greater than 1 signifies the athletes performed better in their home country.

To determine if the medal rate ratio is different between countries, a new transformation parameter for the difference between the medal rate ratio of country *i* and country *j*, i.e. $\Delta r_{ij} = r_i - r_j$, is defined. Utilizing the 100,000 MC samples utilized to construct the posterior distributions, equal-tailed 95% credible intervals are constructed for Δr_{ij} . If the medal rate ratio is different between country *i* and country *j*, then this interval should not contain 0. The countries that were found to differ from one another are those listed in Figure 4.



Figure 4: Posterior distributions of $r_i = \lambda_{i1}/\lambda_{i0}$ and countries that show a difference in r_i (right)

Conclusions

From the analysis performed in this report, the question of if there is a home-country advantage was investigated. From this study it was found that from the aggregated data, there is only a 0.506% chance that the true value of λ_1 is greater than the true value for λ_0 . When a similar analysis was done on a country-by-country basis, there is an observed difference with countries performing significantly better than other countries when they host the Olympics. However, the majority of the countries analyzed in this study did not show significant differences from each other or from the year they hosted and the previous year. When attempting to predict the number of medals with uncertainty France will win in 2024 when they host, a PPD was constructed and used to show that there is a 95% probability that France will win between 26-72 medals in 2024 with a mean medal count of 46.64. Two limitations of this analysis are (1) the number of predictors and (2) not having a detailed breakdown of the events being held at each Olympics. For future work, more predictors can be used, such as country population or country GDP and the specific sports at each Olympics can be further explored since host countries tend to add sports that are popular in their country.

Andy Rivas

```
setwd('C:/Users/aknri/OneDrive/Desktop/Applied Bayesian Stats/Midterm1/')
11
12
13
14 medals <- read.csv("Medals.csv")</pre>
15 attach(medals)
16 set.seed(1)
18 - #####
19 # a #
20 • #####
22 NO = sum(medals$PARTICIPATING.ATHLETES.DURING.PREVIOUS.OLYMPICS)
   N1 = sum(medals$PARTICIPATING.ATHLETES.DURING.HOST.YEAR)
    Y0 = sum(medals$MEDALS.WON.DURING.PREVIOUS.OLYMPICS)
    Y1 = sum(medals$MEDALS.WON.DURING.HOST.YEAR)
    lambda0 = Y0/N0
    lambda1 = Y1/N1
   a = 0.1
    b = 0.1
    A0 = Y0 + a
36 B0 = N0 + b
    mean_lambda_0 = A0/B0
38 variance_lambda_0 = A0/B0^2
39 median_lambda_0 = qgamma(0.50, Y0+a,N0+b)
    upper_bound_lambda_0 = qgamma(0.975, Y0+a,N0+b)
41 lower_bound_lambda_0 = qgamma(0.025, Y0+a,N0+b)
   A1 = Y1 + a
    B1 = N1 + b
    mean_lambda_1 = A1/B1
    variance_lambda_1 = A1/B1^2
    wai rance_rambad_1 = qgamma(0.50, Y1+a,N1+b)
upper_bound_lambda_1 = qgamma(0.975, Y1+a,N1+b)
lower_bound_lambda_1 = qgamma(0.025, Y1+a,N1+b)
```

<pre>2 # ###CTURESS 0.1.1001_01_01_01_01_01_01_01_01_01_01_01_01</pre>	51	
<pre>/ restriction [doi: 10000] / restriction [d</pre>	52	# show results in a table
<pre>/************************************</pre>	53	result_table_a = round(matrix(c(mean_lambda_0,variance_lambda_0,median_lambda_0,lower_bound_lambda_0,upper_bound_lambda_0,mean_lambda_1, variance_lambda_1, upper_bound_lambda_0,upper_bound_lambda_0,mean_lambda_1
<pre>c classificable.ledic_j < cf equations [displance].model.model.pits [displance].model [displance].mo</pre>	54	, var rance_ramoda_r, medrar_ramoda_r, rower_pound_ramoda_r, upper_pound_ramoda_r, in ow=2, incor=3, byrow=1),) rownames(result table a) < - (c'lambda 0''' lambda 1'')
<pre>pretro result, rest, rest</pre>		colnames(result_table_a) <- c("Mean", "Variance", "Median", "95% Lower Bound", "95% Uppper Bound")
<pre>/ Profit Table J / Profi</pre>	56	result_table_a<- as.table(result_table_a)
<pre>procession for poster / set / poster / set / poster / poster</pre>	57 58	result_table_a
<pre>b label_a_sc(_0, i, 0, i, 0, 0, i) protect [</pre>	59	
<pre>protect provide the second provide and the second provide the sec</pre>		lambda = seq(0, 0.3, 0.0001)
<pre>prove the set of the set of</pre>	51	posterior_non_host = dgamma(lambda,Y0+a,N0+b)
<pre>bit inter (unbody poster (in play = , inter), inter), inter), inter), inter), inter), inter((unbody poster (in play = , inter), inter), inter), inter), inter), inter), inter((unbody poster (inter), inter), int</pre>	52 53	poster for instant = ugamma(ramova, irra, $n_1+\alpha$, $n_1+\alpha$) plot(lambda, posterior_non host.col="red". type="l".lwd=3. xlab="lambda". vlab="PDF". cex.lab = 1.5. cex.axis = 1.5. vlim = c(0.100))
<pre>3 legen(tcp)def*_legen(c(tp)=.prod(c(tp), host), col=(trp), host(), p), dec((tp)), dec((tp)),</pre>		lines(lambda,posterior_host,col="blue", lwd=3, xlab="lambda", ylab="PDF", cex.lab = 1.5, cex.axis = 1.5)
<pre>prefix prefix pref</pre>	55	legend("topleft",legend=c("Non-Host","Host"),col=c("red","blue"),lwd=c(3,3), cex=1.5, pt.cex = 1.5)
<pre>pref = pref = pref</pre>	50 57 🗸	
<pre>- #### 5 = 10000 / program if there is hown country identize / program (if there is hown country identize using different priors / i = 1 / i</pre>	58	
<pre>s = number = num</pre>	59 🔻	#####
<pre>2 / pictoring (f the's is inverse country defaultage // pictoring man.lambd.l.w.c. r spamma(S, Vis, Noth) man.lambd.l.w.c. r spamma(S, Vis, Noth) man.lambd.l.w.c. r spamma(S, Vis, Noth) man.lambd.l.m.d.g. r spamma(S, Vis, Noth) man.lambd.l.g. r spamma(S, Vis, N</pre>	/0 71	S = 100000
<pre>gr/model gr/model market_larger_comment(cyto_stoch) market_larger_com</pre>	72	
<pre>imdot_lsc < rpsm(s,vt=,vd=s) imdot_lsc <</pre>	73	# Prior 1
<pre>mean_labds_1 = (yo-a)/(who) proceedings = (</pre>	/4 75	lambda_0_MC <- rgamma(S,YU+a,NU+b) Jambda 1 MC <- rgamma(S YI+a NI+b)
<pre>f mex.labds_0 = (0-1)((0-b) upper_bond_labds_0 = (upun(0-2); v0-1, upb) upper_bond_labds_0 = (upun(0-2); v0-1, upb) upper_bond_labds_0 = (upun(0-2); v1-1, u1-b) upper_bond_labds_0 = (upun(0-2); v1</pre>	76	
<pre>upper_bound_lambd_0 = quant(0, 02, Vol.A00b) liver_bound_lambd_0 = quant(0, 02, Vol.A00b) imer_bound_lambd_0 = quant(0, 02, Vol.A00b) imer_bound_lambd_1 = quant(0, 02, Vol.A00b) imer_bound_lamb</pre>		mean_lambda_0 = (Y0+a)/(N0+b)
<pre>Note Loom2_ manu2 > quant(c)::::::::::::::::::::::::::::::::::::</pre>	78	upper_bound_lambda_0 = ggamma(0.975, Y0+a,N0+b)
<pre>1 mean.labds_1 = (r1-a)((ki-b)</pre>	30	Tower_bound_Tambua_0 = qgamma(0.02), TO+a, NO+D
<pre>g upper_bound_lambdi_1 = quama(0.025, V1+4, H-H) f texer_bound_lambdi_1 = quama(0.025, V1+2, H-H) f texer_bound_lambdi_1 = quama(0</pre>		$mean_lambda_1 = (Y1+a)/(N1+b)$
<pre>inter_point_instruct_ = quant(0.05; vis1, http:// processing.com/ process</pre>	32	upper_bound_lambda_1 = qgamma(0.975, Y1+a,N1+b)
<pre>5 / Detroine if there is a how country docatage using different priors</pre>	33 34	Tower_bound_tamova_t = qyamma(0.022, 11+a, NL+D)
<pre>6 # / i = 1</pre>	35	
<pre>f H = 1 labda_D_W_1 <= rgama(5, Y0+1, N0+b1) labda_L_W_L <= rgama(5, Y0+1, N0+b1) labda_L_W_L <= rgama(5, Y0+1, N0+b1) labda_L_W_L <= rgama(5, Y0+1, N0+b1) labda_L_W_L == rgama(5, Y0+1, N0+b1) labda_L_H_L == rgama(5, Y0+2, N0+b2) labda_L_H_L == rgama(5, Y0+2, Y0+</pre>	36	# Prior 2
<pre>plands_0_MC_1 <- rgama(S, Y0+31, M0+b1) limb(_0_L_C, <- rgama(S, Y0+31, M0+b1) limb(_0_L, <- rgama(S, Y0+32, M0+b2) limb(_0_L, <- rgama(S, Y0+32, M0+b3) limb(_0_L, <- rgama(S, Y0+32, M0</pre>	57 38	
<pre>0 labda_LWC_1 <> rights(s, Yi-ai, Wi-bi) rear_labda_L = (yi-ai)/(Wo-bi) rear_labda_L = (yi-ai)/(Wo-bi) rear_labda_L = (yi-ai)/(Wo-bi) rear_labda_L = (yi-ai)/(Wi-bi) rear_labda_L =</pre>	39	Jambda_0_Mc_1 <- rgamma(5,Y0+a1,N0+b1)
<pre>// seal_labdd_0_l = (0;sl)/(0;bl) import_bourd_labdd_0_l = dpamma(0;0;;, 0;sl,0;bl) import_bourd_labdd_1_l = dpamma(0;0;;, 0;sl,0;bl) import_bourd_labdd_0_l = dpamma(0;0;;, 0;sl,0;bl) import_bourd_labdd_0_l = dpamma(0;0;;, 0;sl,0;bl) import_bourd_labdd_0_l = (qpamma(0;0;;, 0;sl,0;bl) import_bourd_labdd_0_l = dpamma(0;0;;, 0;sl,0;bl) import</pre>	90	lambda_1_MC_1 <- rgamma(s, y1+a1, N1+b1)
<pre>5 mpper bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1,Vi=1,Vi=1) 1 fower_bound_lambd_lime(s,Vi=2,Vi=1,Vi=1,Vi=1,Vi=1,Vi=1,Vi=1,Vi=1,Vi=1</pre>	91 32	moon lambda 0.1 = $(Y0+31)/(Y0+b1)$
<pre>4 Toker_bourd_lambda_l_1 = (qiama(0,02), v0:41, Nobb) 5 mean_lambda_l_1 = (qiama(0,02), v1:41, Nibb) 6 mean_lambda_l_1 = (qiama(0,02), v1:41, Nibb) 6 mean_lambda_l_1 = (qiama(0,02), v1:41, Nibb) 6 mean_lambda_l_2 = (qiama(0,02), v1:41, Nibb) 7 mean_lambda_l_2 = (qiama(0,02), v1:41, Nibb) 7 mean_lambda_l_2 = (qiama(0,02), v1:41, Nibb) 7 mean_lambda_l_3 = (qiama(0,02), v1:41, Nibb) 7 mean_lambda</pre>	93	upper_bound_lambd_0.1 = gamma(0.975, Y0+a1,N0+b1)
<pre>prepr_bourd_labds_l_1 = (yl.al/(yl.th)) prepr_bourd_labds_l_1 = (gamad(0.025, yl.al,Nl.tb)) prepr_bourd_labds_l_1 = (gamad(0.025, yl.al,Nl.tb)) // // // // // // // // // // // // //</pre>	94	lower_bound_lambda_0_1 = qgamma(0.025, Y0+a1,N0+b1)
<pre>prover_bound_lambda_lil = qgama(0,025; v1+a1,v1+b1) to Tower_bound_lambda_lil = qgama(0,025; v1+a1,v1+b1) to Tower_bound_lambda_lil = qgama(0,025; v1+a1,v1+b1) to Tower_bound_lambda_lil = qgama(0,025; v1+a1,v1+b1) tower_bound_lambda_lil = qgama(0,025; v1+a2,v1+b2) tower_bound_lambda_lil = qgama(0,025; v1+a3,v1+b3) tower_bound_lambda_lil = qgama(0,025; v1+a</pre>	95 96	mean lambda 1 1 = $(Y1+a1)/(N1+b1)$
<pre>% Tower_bound_lambda_l_l = qgamma(0:025, V1+a1,N1+b1) % % % % % % % % % % % % % % % % % % %</pre>		upper_bound_lambda_1_1 = qgamma(0.975, Y1+a1,N1+b1)
<pre>// /// // // // // // // // // // // //</pre>	98	Tower_bound_Tambda_1_1 = qgamma(0.025, Y1+a1,N1+b1)
<pre>99 / #Prior i 00 z = 2 00 z =</pre>		
<pre>00 #Prior 3</pre>	0.0	
<pre>provide the set of the set o</pre>		
<pre>101 landdUK_2 < rgamma(S, Vi+2, NO+b2) landdUK_2 < rgamma(S, Vi+2, NO+b2) mean_landb_0_2 = (Vi+2)/(N0+b2) lower_bound_landb_0_2 = gamma(0.025, Vi+2, NO+b2) lower_bound_landb_0_2 = gamma(0.025, Vi+2, NO+b2) lower_bound_landb_0_2 = (gamma(0.025, Vi+2, NI+b2) lower_bound_landb_0_2 = (gamma(0.025, VI+2, NI+b2) lower_bound_landb_0_2 = (gamma(0.025, VI+2, NI+b2) lower_bound_landb_0_2 = (gamma(0.025, VI+2, NI+b2) lower_bound_landb_0_2 = (gamma(0.025, VI+2, NI+b2) lower_bound_landb_0_3 = (gamma(0.025, VI+2, NI+b2) lower_bound_landb_0_3 = (gamma(0.025, VI+2, NI+b2) lower_bound_landb_0_3 = (gamma(0.025, VI+2, NI+b3) lower_bound_landb_0_3 = (gamma(0.025, VI+2, NI+b3) lower_bound_landb_0_3 = (gamma(0.025, VI+2, NI+b3) lower_bound_landb_0_3 = (gamma(0.025, VI+2, NI+b3) lower_bound_landb_0_3 = (gamma(0.025, VI+2, NI+b3) lower_bound_landb_0_1 = (gamma(0.025, VI+2, VI</pre>	99 100	# Prior 3
<pre>10 Imbdal_WC_2 < rgamma(5,1+32,Ni+b2) 10 mer_bond_lambda_0_2 = (quamma(5,2); (V+32,N0+b2) 10 mer_bond_lambda_0_2 = (quamma(5,2); (V+32,N0+b2) 11 upper_bond_lambda_1_2 = (quamma(5,2); (V+32,N0+b2) 12 mer_bond_lambda_1_2 = (quamma(5,2); (V+32,N1+b2) 13 mer_bond_lambda_1_2 = (quamma(5,2); (V+32,N1+b2) 14 mer_bond_lambda_1_2 = (quamma(5,2); (V+32,N1+b2) 14 mer_bond_lambda_0_3 = (V+32),(N0+b3) 14 mer_bond_lambda_0_3 = (V+32),(N0+b3) 15 mer_bond_lambda_0_3 = (quamma(5,2); (V+32,N0+b3) 15 mer_bond_lambda_0_4 = (quamma(5,2); (V+32,N0+b3) 16 mer_bond_lambda_0_3 = (V+32),(N0+b3) 17 mer_bond_lambda_0_3 = (quamma(5,2); (V+33,N0+b3) 18 mer_lambda_1_3 = (quamma(5,2); (V+33,N0+b3) 19 mer_bond_lambda_0_3 = (quamma(5,2); (V+33,N0+b3) 10 merabola; (mor_bold_1ambda_0,mor_bold_1ambda_</pre>	99 100 101 102	# Prior 3 a2 = 2 b2 = 2
<pre>mean_labda_0_2 = (Y0+22)/(N0+b2) upper_bound_labda_0_2 = qgama(0.025; Y0+a2,N0+b2) mean_labda_1_2 = (Y1+31)/(N1+b1) upper_bound_labda_1_2 = qgama(0.025; Y1+a2,N1+b2) imwer_bound_labda_1_2 = qgama(0.025; Y1+a2,N1+b2) imwer_bound_labda_1_2 = qgama(0.025; Y1+a2,N1+b2) imwer_bound_labda_1_2 = qgama(0.025; Y1+a2,N1+b2) imwer_bound_labda_1_2 = qgama(0.025; Y1+a2,N1+b2) imwer_bound_labda_1_3 = qgama(0.025; Y1+a2,N1+b2) imwer_bound_labda_1_3 = (Y1+a3)/(N0+b3) imwer_bound_labda_1_3 = qgama(0.025; Y1+a3,N1+b3) imwer_bound_labda_1_3 = qgama(0.025; Y1+a3,N1+b3) imwer_bound_labda_0_0_1_2, pice(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1</pre>	99 100 101 102 103	# Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2)
<pre>10 upper_bound_lambda_0_2 = qgama(0,975, Vo+2,N0+b2) 10 lower_bound_lambda_0_2 = qgama(0,975, Vo+2,N0+b2) 10 mean_lambda_1_2 = (Y1+a1)/(N1+b1) 11 lower_lound_lambda_1_2 = qgama(0,975, V1+2,N1+b2) 11 lower_lound_lambda_1_2 = qgama(0,975, V1+2,N1+b2) 11 lower_lound_lambda_1_2 = qgama(0,975, V1+2,N1+b2) 11 lambda_0_K_3 < rgama(5,V0+3,N0+b3) 11 lambda_0_K_3 < rgama(5,V0+3,N0+b3) 12 lower_lound_lambda_0_3 = (Y0+3)/(N1+b3) 12 upper_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 13 lambda_1_K_3 < rgama(5,V0+3,N0+b3) 14 mean_lambda_0_3 = (Y0+3)/(N1+b3) 15 upper_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 16 lower_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 17 upper_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 18 mean_lambda_0_3 = (Y1+33)/(N1+b3) 19 lower_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 10 lower_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 11 upper_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 12 lower_bound_lambda_1_3 = qgama(0,975, V0+2,N0+b3) 13 result_table_b < (Y1+33)/(N1+b3) 14 result_table_b < (Y1+33)/(N1+b3) 15 result_table_b < (Y1+33)/(N1+b3) 16 result_table_b < (Y1+33)/(N1+b3) 17 result_table_b < (Y1+33)/(N1+b3) 18 result_table_b < (Y1+33)/(N1+b3) 19 result_table_b < (Y1+33)/(N1+b3) 19 result_table_b < (Y1+33)/(N1+b3) 19 result_table_b < (Y1+33)/(N1+b3) 10 result_table_b < (Y1+33)/(N1+b3) 10 result_table_b < (Y1+33)/(N1+b3) 10 result_table_b < (Y1+33)/(N1+b3) 10 result_table_b < (Y1+33)/(N1+b3)/(Y</pre>	99 100 101 102 103 104 105	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2)</pre>
<pre>mem_lambda_l_2 = (y1+al)((x1+b)) mem_lound_lambda_l_2 = (y1+al)(x1+b) lower_bound_lambda_l_2 = (y1+al)(x1+b) lower_bound_lambda_l_2 = (y1+al)(x1+b) lower_bound_lambda_l_2 = (y1+al)(x1+b) lower_bound_lambda_l_2 = (y1+al)(x1+b) lambda_l_9C_3 < (rganma(5, y1+a), x1+b) lambda_l_9C_3 < (rganma(5, y1+a), x1+b) lambda_l_9C_3 < (rganma(5, y1+a), x1+b) lambda_l_3C_3 < (rganma(5, y1+a), x1+b) lambda_l_3C_3 < (rganma(5, y1+a), x1+b) lambda_l_3C_3 < (rganma(5, y1+a), x1+b) lambda_l_3C_3 < (rganma(5, y2+a), x1+b) lower_bound_lambda_l_3 = (ganma(0, 025, y1+a), y1+b) lower_bound_lambda_l_3 = (ganma(0, 025, y1+a), y1+b</pre>	99 100 101 102 103 104 105 106	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2)</pre>
<pre>110 mean_lambda_l_2 = (Y++1)/(Y++10) upper_bound_lambda_l_2 = qgamma(0.925, Y1+a2,N1+b2) 111 lower_bound_lambda_l_2 = qgamma(0.925, Y1+a2,N1+b2) 112 lambda_l_9C_3 < rgamma(s,Y0+a3,N0+b3) 112 lambda_l_9C_3 < rgamma(s,Y0+a3,N0+b3) 112 mean_lambda_l_3 = (Y0+a3)/(N0+b3) 112 mean_lambda_l_3 = (Y0+a3)/(N0+b3) 113 mean_lambda_l_3 = (Y0+a3)/(N1+b3) 114 mean_lambda_l_3 = qgamma(0.925, Y1+a3,N0+b3) 115 upper_bound_lambda_l_3 = qgamma(0.925, Y1+a3,N1+b3) 116 nower_bound_lambda_l_3 = qgamma(0.925, Y1+a3,N1+b3) 117 colnames(result_table_b = round(matrix(C(a,b,100*mean(lambda_l_MC_3)ambda_0_MC_3)), nrow=1, ncol=3, byrow=1),3) 118 result_table_b < < c(Prior_l', Prior_2', Prior_3', Prior_4') 119 result_table_b < < c(Prior_l', Prior_2', Prior_3', Prior_4') 110 result_table_b < < c(Prior_l', Prior_2', Prior_3', Prior_4') 110 result_table_b < < c(Prior_l', Prior_3', Prior_3', Prior_4') 111 result_table_b < < as.table(result_table_b) 112 result_table_b < < as.table(result_table_b) 113 result_table_b_0 < c(Prior_l', Prior_3', Prior_3', Prior_4') 113 result_table_b_0 < c(Prior_l', Prior_3', Prior_3', Prior_4') 114 result_table_b_0 < c(Prior_l', Prior_3', Prior_3', Prior_4') 115 result_table_b_0 < c(Prior_l', Prior_3', Prior_3', Prior_4') 116 result_table_b_0 < c(Prior_l', Prior_3', Prior_3', Prior_4') 117 result_table_b_0 < c(Prior_1', Prior_3', Prior_3', Prior_4') 118 result_table_b_0 < c(Prior_1', Prior_3', Prior_3', Prior_4') 119 result_table_b_0 < c(Prior_1', Prior_3', Prior_3', Prior_4') 119 result_table_b_0 < c(Prior_1', Prior_3', Prior_3', Prior_4') 110 result_table_b_0 < c(Prior_1', Prior_3', Prior_3', Prior_4') 110 result_table_b_0 < c(Prior_1', Prior_3', Prior_3', Prior_4') 110 resu</pre>	99 100 101 102 103 104 105 106 107	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0_2 = ggamma(0.975, Y0+a2,N0+b2) lawnon bound_lambda_0_0_2 = ggamma(0.975, Y0+a2,N0+b2) lawnon bound_lambda_0_0_0_0_0_0_0_0_0_0_0_0_0_0_0_0_0_0_0</pre>
<pre>111 upper_Dound_lambda_l_2 = qgamma(0,025, Y1+32,NL+02) 1 lower_Dound_lambda_l_2 = qgamma(0,025, Y1+32,NL+02) 112 lower_Dound_lambda_l_2 = qgamma(0,025, Y1+32,NL+02) 113 a f a = b 114 a = b 1</pre>	99 100 101 102 103 104 105 106 107 108 109	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0_2 = qgamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_0_2 = qgamma(0.025, Y0+a2,N0+b2)</pre>
<pre>113 whit submits interface again (ote), this prive y 114 / Prior 4 33 = 5 115 3a = 5 115 3a = 5 117 lambda_0.s < rgamma(s, y0+a3, N0+b3) 118 lambda_U.S. 3 < (rgamma(s, y1+a3, N1+b3) 119 110 mean_lambda_0.a = (y0+a3)/(N0+b3) 111 upper_bound_lambda_0.a = (y0+a3)/(N0+b3) 112 upper_bound_lambda_0.a = (y0+a3)/(N0+b3) 112 upper_bound_lambda_0.a = (y0+a3)/(N0+b3) 112 upper_bound_lambda_1.a = (y1+a3)/(N1+b3) 112 upper_bound_lambda_1.a = (y1+a3)/(N1+b3) 113 mean_lambda_1.a = (y1+a3)/(N1+b3) 114 mean_lambda_1.a = (y1+a3)/(N1+b3) 115 weights in a tables 115 weights in a tables 116 lower_bound_lambda_0.mc_2), a3, b3, 100 mean(lambda_1_MC_3) ambda_0_MC), a1, b1, 100 mean(lambda_1_MC_1) ambda_0_MC_1), a2, b2, 100*mean 118 (lambda_1_MC_2) lambda_0_MC_2), a3, b3, 100 mean(lambda_1_MC_3) lambda_0_MC_3), nrow-4, ncol=3, byrow=1), 3 119 result_table_b = round(matrix(c(a, b, mean_lambda_0, upper_bound_lambda_0, c_1)wer_bound_lambda_0_3, a1, b1, mean_lambda_0_1, upper_bound_lambda_0_2, nore-1, byrow=1), 3 113 result_table_b.g = round(matrix(c(a, b, mean_lambda_0, upper_bound_lambda_0_2, nore-1, burd_1_ambda_0_3, upper_bound_lambda_0_2, nore-1, burd_1_ambda_0_2, nore-1, burd_1_ambda_0_3, upper_bound_lambda_0_2, nore-1, burd_1_ambda_0_2, nore-1, burd_1_ambda_0_3, upper_bound_lambda_0_3, upper_bound_lambda_1_3, upper_bound_</pre>	99 100 101 102 103 104 105 106 107 108 109 110	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0_2 = qgamma(0.025, Y0+a2,N0+b2) lower_bound_lambda_0_2 = qgamma(0.025, Y0+a2,N0+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) mean_lambda_1_2 = (Y1+a1)/(N1+b1)</pre>
<pre>114 # Prior 4 115 a3 = 5 15 a3 = 5 16 b3 = 5 17 lambda_0.MC_3 <- rgamma(s, Y0+a3, N0+b3) 18 lambda_0.MC_3 <- rgamma(s, Y1+a3, N1+b3) 19 mean_lambda_0.3 = (qyama(0.925, Y0+a3, N0+b3) 10 mean_lambda_0.3 = (qyama(0.925, Y0+a3, N0+b3) 10 mean_lambda_1.3 = (qyama(0.925, Y1+a3, N1+b3) 116 lower_bound_lambda_1.3 = (qyama(0.925, Y1+a3, N1+b3) 117 mean_lambda_1.4 = (pyama(0.925, Y1+a3, N1+b3) 118 / Show results in a tables 119 result_table_b <- (c^{(n+1)}, (ryintor_2, ryintor_3, "prior_3," prior_3," prior_4) 110 colnames(result_table_b) <- (c((n+1), ryintor_2, ryintor_3," prior_3," prior_4) 111 colnames(result_table_b) <- (c((n+1), ryintor_3, "prior_3," prior_4)) 112 result_table_b.0 = round(matrix(c(a, b, mean_lambda_0, upper_bound_lambda_0, 0, lower_bound_lambda_0, a1, b1, mean_lambda_0, a1, upper_bound_lambda_0, a1, upper_bound_lambda_1, upper_bound_lambda_1, upper_bound_lambda_1, upper_bound_lambda_1, upper_bound_lambda_1, a1, b1, mean_lambda_1, upper_bound_lambda_1, upper_bound_lambda_1, a1, b1, mean_lambda_1, upper_bound_lambda_1, upper_bound</pre>	99 100 101 102 103 104 105 106 107 108 109 110 111 112	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_wC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_wC_2 <- rgamma(5,Y1+a2,N1+b2) upper_bound_lambda_0_2 = (ggamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_0_2 = qgamma(0.025, Y0+a2,N0+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qgamma(0.975, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.975, Y1+a2,N1+b2)</pre>
<pre>bit dis</pre>	99 100 101 102 103 104 105 106 107 108 109 110 111 112 113	<pre>/# Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0_2 = qgamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_0_2 = qgamma(0.025, Y0+a2,N0+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qgamma(0.975, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2)</pre>
<pre>117 lambda_0_K_2 <- rgamma(S,Y0+a3,N0+b3) 118 lambda_1_K_2 rgamma(S,Y0+a3,N0+b3) 119 119 exan.lambda_0_3 = (Y0+a3)/(N0+b3) 111 upper_bound_lambda_0_3 = qgamma(0.975, Y0+a3,N0+b3) 112 upper_bound_lambda_0_3 = qgamma(0.975, Y0+a3,N0+b3) 112 upper_bound_lambda_1_3 = qgamma(0.925, Y0+a3,N0+b3) 112 upper_bound_lambda_1_3 = qgamma(0.925, Y0+a3,N0+b3) 112 upper_bound_lambda_1_3 = qgamma(0.925, Y1+a3,N1+b3) 112 upper_bound_lambda_1_3 = qgamma(0.925, Y1+a3,N1+b3) 113 lower_bound_lambda_1_3 = qgamma(0.925, Y1+a3,N1+b3) 114 lower_bound_lambda_1_3 = qgamma(0.925, Y1+a3,N1+b3) 115 lower_bound_lambda_1_3 = qgamma(0.925, Y1+a3,N1+b3) 116 lower_bound_lambda_1_3 = qgamma(0.925, Y1+a3,N1+b3) 117 / Show results in a tables 117 result_table_b = round(matrix(c(a,b,100*mean(lambda_1_MC)lambda_0_MC_3), nrow=4, ncol=3, byrow=1),3) 118 result_table_b = round(matrix(c(a,b,mean(lambda_1_MC)lambda_0_MC_3), nrow=4, ncol=3, byrow=1),3) 119 result_table_b <- (C(*a', b', *rob')) 120 result_table_b as.table(result_table_b) 121 result_table_b as.table(result_table_b) 122 result_table_b (C(*a', b', *rob')) 123 rownames(result_table_b) <- (C(*a', b', *rob')) 124 rownames(result_table_b) <- (C(*a', b', *rob')) 125 rownames(result_table_b) <- (C(*a', b', *rob')) 126 rownames(result_table_b) <- (C(*a', b', *rob')) 127 result_table_b as.table(result_table_b) 128 result_table_b as.table(result_table_b) 129 result_table_b as.table(result_table_b) 120 rownames(result_table_b) <- (C(*a', b', *roa'), *roin_3', *roin_4') 121 rownames(result_table_b) <- (C(*a', b', *roa'), *roin_3', *roin_4') 122 rownames(result_table_b) <- (C(*a', b', *roa'), *roin_3', *roin_4') 123 rownames(result_table_b) <- (C(*a', b', *roa'), *roin_3', *roin_4') 124 result_table_b_1 <- round(matrix(c(a, b,mean_lambda_1, upper_bound_lambda_1, ai, bi,mean_lambda_1, upper_bound_lambda_1, ai, bi,mean_lambda_1, upper_bound_lambda_1, ai, bi,mean_lambda_1, upper_bound_lambda_1, ai, vomer_bound_lambda_1, ai, vomer_bound_lambda_1, ai, vomer_bound_lambda_1, ai, vomer_bo</pre>	999 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114	<pre>// Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rganma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rganma(5,Y1+a2,N1+b2) mean_lambda_0_2 = qganma(0.075, Y0+a2,N0+b2) lower_bound_lambda_0_2 = qganma(0.025, Y0+a2,N0+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2)</pre>
<pre>118 1ambda_LMC_3 <- rgamma(s,Yl+3,NL+b3) 129 120 mean_lambda_0.3 = (Y0+33)/(NL+b3) 121 upper_bound_lambda_0.3 = qgamma(0.025, Y0+3,N0+b3) 122 lower_bound_lambda_0.3 = qgamma(0.025, Y0+3,N0+b3) 123 mean_lambda_1.3 = (Y1+33)/(NL+b3) 124 mean_lambda_1.3 = (Y1+33)/(NL+b3) 125 upper_bound_lambda_1.3 = (qgamma(0.025, Y1+3,NL+b3) 126 lower_bound_lambda_1.3 = (qgamma(0.025, Y1+3,NL+b3) 127 128 <i>f show results in a tables</i> 129 result_table_b = round(matrix(c(a,b,100*mean(lambda_1_MC_)anbbda_0_MC_),a1,b1,100*mean(lambda_1_MC_1)ambda_0_MC_1),a2,b2,100*mean (lambda_LMC_2)aa,b3,100*mean(lambda_LMC_3)anbbda_0_MC_3), nrow=4, ncol=3, byrow=1),3) 120 rownames(result_table_b) << c(?a', 'b', 'prob') 120 result_table_b <- as.table(result_table_b) 135 result_table_b <- c(?a', 'b', 'prob'),3) 136 rownames(result_table_b) << c(?a', 'b', 'pror.2', 'prior.2', 'prior</pre>	999 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5, Y0+a2, N0+b2) lambda_1_MC_2 <- rgamma(5, Y1+a2, N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0_2 = qgamma(0.975, Y0+a2, N0+b2) lower_bound_lambda_0_2 = qgamma(0.025, Y0+a2, N0+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qgamma(0.975, Y1+a2, N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2, N1+b2) # Prior 4 a3 = 5 b3 = 5</pre>
<pre>mean_lambda_0_3 = (Y0+a3)/(N0+b3) upper_bound_lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3) line=pound_lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3) line=pound_lambda_1_3 = (Y1+a3)/(N1+b3) upper_bound_lambda_1_3 = qgamma(0.025, Y1+a3,N1+b3) line=pound_lambda_1_3 = qgamma(0.025, Y1+a3,N1+b3) line=pound_lambda_1_3 = qgamma(0.025, Y1+a3,N1+b3) line=pound_lambda_1_3 = qgamma(0.025, Y1+a3,N1+b3) line=pound_lambda_1_3 = qgamma(0.025, Y1+a3,N1+b3) line=pound_lambda_1_S = qgamma(0.025, Y1+a3,N1+b3) line=pound_lambda_0_S = qgamma(0, quantile (2.55)", "quantile (97,55)") result_table_b_0 <- c(*1+,*0+,*0+,*0+,*0+,*0+,*0+,*0+,*0+,*0+,*0</pre>	999 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117	<pre>/# Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(S,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(S,Y1+a2,N1+b2) mean_lambda_0_2 = (y0+a2)/(N0+b2) upper_bound_lambda_0_2 = qgamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y0+a2,N0+b2) mean_lambda_1_2 = (y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qgamma(0.975, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) /# Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(S,Y0+a3,N0+b3)</pre>
<pre>11 upper_bound_lambda_0_3 = qgamma(0.975, Y0+a3,N0+b3) 12 lower_bound_lambda_0_3 = qgamma(0.975, Y0+a3,N0+b3) 13 mean_lambda_1_3 = (Y1+a3)/(N1+b3) 13 upper_bound_lambda_1_3 = qgamma(0.975, Y1+a3,N1+b3) 14 mean_lambda_1_3 = qgamma(0.025, Y1+a3,N1+b3) 15 lower_bound_lambda_1_3 = qgamma(0.025, Y1+a3,N1+b3) 16 lower_bound_lambda_1_3 = ggamma(0.025, Y1+a3,N1+b3) 17 result_table_b = round(matrix(c(a,b,100*mean(lambda_1_MC>lambda_0_MC),a1,b1,100*mean(lambda_1_MC_1>lambda_0_MC_1),a2,b2,100*mean 17 (lambda_1_MC_2>lambda_0_MC_2),a3,b3,100*mean(lambda_1_MC_3>lambda_0_MC_3), nrow=4, ncol=3, byrow=T),3) 18 rownames(result_table_b) < c("arin", "Prior_2", "Prior_3", "Prior_4") 19 result_table_b < - as.table(result_table_b) 10 rownames(result_table_b) < c("arin", "Prior_2", "Prior_3", "Prior_4") 10 rownames(result_table_b.0 < c("arin", "Prior_2", "Prior_3", "Prior_4") 11 rownames(result_table_b.0 < c("arin", "Prior_2", "Prior_3", "Prior_4") 12 rownames(result_table_b.0 < c("arin", "Prior_2", "Prior_3", "Prior_4") 13 result_table_b.0 < as.table(result_table_b) 14 result_table_b.0 < c("arin", "Prior_2", "Prior_3", "Prior_4") 15 rownames(result_table_b.0) < c("arin", "Prior_2", "Prior_3", "Prior_4") 16 rownames(result_table_b.0) < c("arin", "Prior_2", "Prior_3", "Prior_4") 17 rownames(result_table_b.0) < c("arin", "Prior_2", "Prior_3", "Prior_4") 18 result_table_b.0 < c(sa, table(result_table_b.0) 19 result_table_b.0 < c(sa, table(result_table_b.0) 10 rownames(result_table_b.0) < c("arin", "Mean", "Quantile (2.5%)", "Quantile (2.5%)") 10 rownames(result_table_b.0) < c("arin", "Mean", "Quantile (2.5%)", "Quantile (2</pre>	999 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118	<pre>/# Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(S,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(S,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) lower_bound_lambda_0_2 = qgamma(0.025, Y0+a2,N0+b2) lower_bound_lambda_1_2 = (y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) /# Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(S,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(S,Y1+a3,N1+b3)</pre>
<pre>rower_bound_lambda_l_3 = (rl+a3)/(Nl+b3) mean_lambda_l_3 = (rl+a3)/(Nl+b3) mean_lambda_l_4 = round(matrix(c(a,b,100*mean(lambda_l_MC>lambda_0_MC),al,bl,100*mean(lambda_l_MC_1>lambda_0_MC_1),a2,b2,100*mean (lambda_l_MC_2>lambda_0_MC_2),a3,b3,100*mean(lambda_l_MC_3>lambda_0_MC_3)), nrow=4, ncol=3, byrow=T),3) rownames(result_table_b) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") rownames(result_table_b) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") result_table_b <- as.table(result_table_b) result_table_b <- as.table(result_table_b) result_table_b <- eround(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0_1,lower_bound_lambda_0_1,ab,b1,mean_lambda_0_3,upper_bound_lambda_0_3,</pre>	999 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 1120	<pre>/# Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) upper_bound_lambda_0_2 = qgamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_0_2 = qgamma(0.025, Y0+a2,N0+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qgamma(0.975, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) # Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(5,Y1+a3,N1+b3)</pre>
<pre>124 mean_lambda_1_3 = (Y1+a3)/(N1+b3) 125 upper_bound_lambda_1_3 = ggamma(0.975, Y1+a3,N1+b3) 126 lower_bound_lambda_1_3 = ggamma(0.025, Y1+a3,N1+b3) 127 128 / Show results in a tables 129 result_table_b = round(matrix(c(a,b,100*mean(lambda_1_MC_3>lambda_0_MC,3), nrow=4, ncol=3, byrow=T),3) 120 rownames(result_table_b) <- c("prior_1", "prior_2", "prior_3", "prior_4") 131 colnames(result_table_b) <- c("a", "b", "prior_2", "prior_3", "prior_4") 132 result_table_b <= round(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0,lower_bound_lambda_0_a,a1,b1,mean_lambda_0_a,upper_bound_lambda_0_a], 133 result_table_b 14 155 result_table_b <= round(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0,lower_bound_lambda_0_a,a1,b1,mean_lambda_0_a,upper_bound_lambda_0_a], 139 rownames(result_table_b) <- c("a", "b", "mean", "Quantile (2.5%)", "Quantile (97.5%)") 130 rownames(result_table_b.0) 131 result_table_b.0 <- a.s.table(result_table_b.0) 132 result_table_b.0 <- c("a", "b", "mean", "Quantile (2.5%)", "Quantile (97.5%)") 133 result_table_b.0 140 140 140 140 141 142 rownames(result_table_b.1) <- c("a", "b", "mean", "Quantile (2.5%)", "Quantile (97.5%)") 143 result_table_b.1 <- c("a", "b", "mean", "Quantile (2.5%)", "Quantile (97.5%)") 144 result_table_b.1 <- c("a", "b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 145 result_table_b.1 146 140 140 140 140 140 140 140 141 15 141 15 15 15 15 15 15 15 15 15 15 15 15 15</pre>	999 1000 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rganma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rganma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) lower_bound_lambda_0_2 = qganma(0.975, Y0+a2,N0+b2) lower_bound_lambda_1_2 = qganma(0.975, Y1+a2,N1+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2) # Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rganma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- rganma(5,Y0+a3,N0+b3) mean_lambda_0_3 = (Y0+a3)/(N0+b3) upper_bound_lambda_0_3 = ggamma(0.975, Y0+a3,N0+b3) lower_bound_lambda_0_3 = ggamma(0.975, Y0+a3,N0+b3)</pre>
<pre>upper_bound_lambda_l_3 = qgamma(0.9/5, Yl+a3,Nl+b3) lower_bound_lambda_l_3 = qgamma(0.025, Yl+a3,Nl+b3) /// # show results in a tables result_table_b = round(matrix(c(a,b,100*mean(lambda_l_MC>lambda_0_MC),al,bl,100*mean(lambda_l_MC_l>lambda_0_MC_l),a2,b2,100*mean (lambda_L_MC_2>lambda_0_MC_2),a3,b3,100*mean(lambda_L_MC_3>lambda_0_MC_3)), nrow=4, ncol=3, byrow=1),3) rownames(result_table_b) <- c("en', "b", "prob") result_table_b.(- c("en', "b", "prob") result_table_b.(- c("a', "b", "prob") result_table_b.(- as.table(result_table_b) result_table_b.(- as.table(result_table_b) rownames(result_table_b.(- as.table(result_table_b) rownames(result_table_b.(- as.table(result_table_b) rownames(result_table_b.(- c("en', "b", "prob")) rownames(result_table_b.(- c("a', "b", "prob")) rownames(result_table_b.(- as.table(result_table_b) rownames(result_table_b.(- as.table(result_table_b)) rownames(result_table_b.(- as.table(result_table_b)) rownames(result_table_b.(- c("a', "b", "prof", "prior_3", "prior_4") rownames(result_table_b.(- c("a', "b", "kean", "Quantile (2.5%)", "Quantile (97.5%)") result_table_b.(- c("a', "b", "kean", "Quantile (2.5%)", "Quantile (97.5%)") result_table_b.(- c("a', "b", "prior_2", "prior_3", "prior_4") rownames(result_table_b.(- c("as.table(result_table_b.(- c("a', "b", "kean", "Quantile (2.5%)", "Quantile (97.5%)") result_table_b.(- c("as.table(result_table_b.(- c("a', "b", "prior_2", "prior_3", "prior_4") rownames(result_table_b.(- c("a', "b", "kean", "Quantile (2.5%)", "Quantile (97.5%)") result_table_b.(- c("as.table(result_table_b.(- c("a', "b", "prior_3", "prior_4") rownames(result_table_b.(- c("a', "b", "prior_2", "prior_3", "prior_4") rownames(result_table_b.(- c("as.table(result_table_b.(- c("a', "b", "prior_3", "prior_4")) rownames(result_table_b.(- c("a', "b", "prior_3", "prior_4") rownames(result_table_b.(- c("a', "b", "prior_3", "prior_4") rownames(result_table_b.(- c("a', "b", "prior_3", "prior_3", "prior_4") rownames(result_table_b.(- c("a', "b", "prior_3", "prior_4")) rownames(r</pre>	999 1000 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 122	<pre>/* Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = qgamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_0_2 = qgamma(0.975, Y0+a2,N0+b2) mean_lambda_1_2 = (Y1+a1)/(N1+b1) upper_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) /* Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(5,Y1+a3,N1+b3) mean_lambda_0_S = (Y0+a3)/(N0+b3) upper_bound_lambda_0_3 = (qgamma(0.975, Y0+a3,N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3)</pre>
<pre>// # show results in a tables result_table_b = round(matrix(c(a,b,100*mean(lambda_1_MC>lambda_0_MC_3), nrow=4, ncol=3, byrow=T),3) rownmes(result_table_b) <- c("ar", "b", "prior_2", "prior_3", "prior_4") colnames(result_table_b) <- c("a", "b", "prob") result_table_b <- a.s.table(result_table_b) result_table_b.0 = round(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0,lower_bound_lambda_0,a1,b1,mean_lambda_0_1,upper_bound_lambda_0_3) rownmes(result_table_b.0) <- c("ar", "b", "prior_3", "prior_4") rownmes(result_table_b.0) <- c("ar", "b", "wean", "quantile (2.5%)", "quantile (97.5%)") result_table_b.0 <- c("a", "b", "wean", "quantile (2.5%)", "quantile (97.5%)") result_table_b.1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1,lower_bound_lambda_1_a,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_dound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_dound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_dound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_3, nrow=4, ncol=5, byrow=1,3) rownames(result_table_b_1) <- c("prior_1", "Prior_2", "Prior_3", "Prior_4") colnames(</pre>	999 1000 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123	<pre>% Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0_2 = qgamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_1_2 = qgamma(0.975, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2,N1+b2) // Prfor 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3,N0+b3)</pre>
<pre>128 # Show results in a tables 129 result_table_b = round(matrix(c(a,b,100*mean(lambda_1_Mc_lambda_0_MC_3)), nrow=4, ncol=3, byrow=T),3) 130 rownames(result_table_b) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 131 colnames(result_table_b) <- c("C"prior_1", "Prior_2", "Prior_3", "Prior_4") 132 result_table_b <- cas.table(result_table_b) 133 result_table_b <- cas.table(result_table_b) 134 result_table_b <- cas.table(result_table_b) 135 result_table_b <- cas.table(result_table_b) 136 rownames(result_table_b) <- c("a", b", "prior_2", "Prior_3", "Prior_4") 137 rownaes(result_table_b) <- c("a", b", "prior_2", "Prior_3", "Prior_4") 138 result_table_b <- cas.table(result_table_b) 139 rownames(result_table_b) <- c("a", b", "mean", "Quantile (2.5%)", "Quantile (97.5%)") 139 result_table_b.0 <- c("a", b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 130 rownames(result_table_b.0 <- c("a", b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 131 result_table_b.1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1,upper_bound_lambda_1,a 137 rownames(result_table_b.0 <- c("a", b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 138 result_table_b.0 <- c("prior_1", "Prior_2", "Prior_3", "Prior_4") 139 result_table_b.0 <- c("a", b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 130 result_table_b.1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1,upper_bound_lambda_1, 1,ower_bound_lambda_1,a2,b2,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a3,b3,mean_lambda_1,upper_bound_lambda_1,a3,b3,mean_lambda_1,a3,b3,mean_lambda_1,a3,b3,mean_lambda_1,a3,upper_bound_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lambda_1,a3,a3,b3,mean_lamb</pre>	999 1000 101 102 103 104 105 106 107 108 109 110 110 110 111 1112 113 114 115 116 117 118 1120 121 122 123 124 125 126	<pre>/* Prior 3 a2 = 2 b2 = 2 lambda_0_XC_2 <- rganma(S,Y0+a2,N0+b2) lambda_1_XC_2 <- rganma(S,Y1+a2,N1+b2) mean_lambda_0_2 = (q0+a2)/(N0+b2) upper_bound_lambda_0_2 = qganma(0.025, Y0+a2,N0+b2) lower_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = qganma(0.025, Y1+a2,N1+b2) // Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rganma(S,Y0+a3,N0+b3) lambda_1_MC_3 <- rganma(S,Y1+a3,N1+b3) mean_lambda_0_3 = qganma(0.075, Y0+a3,N0+b3) lower_bound_lambda_0_3 = qganma(0.075, Y0+a3,N0+b3) lower_bound_lambda_0_3 = qganma(0.075, Y1+a3,N1+b3)</pre>
<pre>result_table_b = round(matrix((ta,b,to),to)=mean(tambda_1_MC_3)tabla_0_MC_3), if b1, now=ean(tambda_1_MC_1)tabla_0_MC_1), a2, b2, 100* mean((lambda_1_MC_2): a3, b3, 100* mean(lambda_1_MC_3)tabla_0_MC_3)), nrow=4, ncol=3, byrow=T), 3) rownames(result_table_b) <- (("Prior_1", "Prior_2", "Prior_3", "Prior_4") colnames(result_table_b) <- (c(a, b, mean_lambda_0, upper_bound_lambda_0, lower_bound_lambda_0, a1, b1, mean_lambda_0_1, upper_bound_lambda_0_1, lower_bound_lambda_0_1, a2, b2, mean_lambda_0_2, upper_bound_lambda_0_2, lower_bound_lambda_0_3, a, b3, mean_lambda_0_3, upper_bound_lambda_0_3 , lower_bound_lambda_0_3, nrow=4, ncol=5, byrow=T), 3) rownames(result_table_b) <- c("a", b", "Prior_2", "Prior_3", "Prior_4") colnames(result_table_b) <- c("a", b", "mean", "quantile (2.5%)", "quantile (97.5%)") result_table_b.0 <- c("a", brow=_lambda_1, upper_bound_lambda_1, lower_bound_lambda_1, a1, b1, mean_lambda_1_1, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1, upper_bound_lambda_1_2, lower_bound_lambda_1_2, a3, b3, mean_lambda_1_3, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1_2, lower_bound_lambda_1_2, a3, b3, mean_lambda_1_3, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1_2, lower_bound_lambda_1_2, a3, b3, mean_lambda_1_3, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1_2, lower_bound_lambda_1_2, a3, b3, mean_lambda_1_3, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1_2, lower_bound_lambda_1_2, a3, b3, mean_lambda_1_3, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1_2, lower_bound_lambda_1_2, a3, b3, mean_lambda_1_3, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1_2, lower_bound_lambda_1_3, upper_bound_lambda_1_3 , lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1_2, upper_bound_lambda</pre>	999 1000 1011 102 103 104 105 106 107 108 109 110 110 111 112 113 114 115 116 117 117 118 119 120 121 122 123 124 125 126 127	<pre>/* Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (y0+a2)/(N0+b2) upper_bound_lambda_0_2 = ggamma(0.025, Y0+a2,N0+b2) lower_bound_lambda_1_2 = (y1+a1)/(N1+b1) upper_bound_lambda_1_2 = (ggamma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = (ggamma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = (ggamma(0.025, Y1+a2,N1+b2) /* Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(5,Y1+a3,N1+b3) mean_lambda_0_3 = (goamma(0.025, Y0+a3,N0+b3) lower_bound_lambda_0_3 = (ggamma(0.025, Y0+a3,N0+b3) lower_bound_lambda_1_3 = ggamma(0.025, Y1+a3,N1+b3) mean_lambda_1_3 = (Y1+a3)/(N1+b3) upper_bound_lambda_1_3 = ggamma(0.025, Y1+a3,N1+b3) lower_bound_lambda_1_3 = ggamma(0.025, Y1+a3,N1+b3)</pre>
<pre>130 rownames(result_table_b) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 131 colnames(result_table_b) <- c("a", b", "prob") 132 result_table_b <- as.table(result_table_b) 133 result_table_b_0 = round(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0,lower_bound_lambda_0,a1,b1,mean_lambda_0_1,upper_bound_lambda_0_3, 1,ower_bound_lambda_0_1,a2,b2,mean_lambda_0_2,upper_bound_lambda_0_2,lower_bound_lambda_0_2,a3,b3,mean_lambda_0_3,upper_bound_lambda_0_3 1,ower_bound_lambda_0_3, nrow=4, ncol=5, byrow=1),3) 139 result_table_b_0 <- c("a", "b", "Prior_2", "Prior_3", "Prior_4") 140 result_table_b_0 <- c("a", "b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 141 result_table_b_1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1_1,upper_bound_lambda_1_3, 1,ower_bound_lambda_1_3, nrow=4, ncol=5, byrow=1),3) 142 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 143 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 144 result_table_b_1 <- c("a", "b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 145 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 145 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 146 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 147 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 148 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 149 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 140 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 141 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 142 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 143 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_4") 144 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_4") 145 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_4") 146 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_4") 146 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_4") 146</pre>	999 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128	<pre># Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5, Y0+a2, N0+b2) lambda_1_MC_2 <- rgamma(5, Y1+a2, N1+b2) mean_lambda_0_2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0_2 = qgamma(0.025, Y0+a2, N0+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2, N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2, N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2, N1+b2) // Prior 4 a3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(5, Y0+a3, N0+b3) lambda_1_MC_3 <- rgamma(5, Y0+a3, N0+b3) lambda_0_MC_3 <- rgamma(0.025, Y0+a3, N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3, N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3, N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y1+a3, N1+b3) // Show results in a tables</pre>
<pre>11 colnames(result_table_b) <- c(a', b', Prob) 2 result_table_b <- as.table(result_table_b) 13 result_table_b_0 = round(matrix(c(a, b, mean_lambda_0, upper_bound_lambda_0, lower_bound_lambda_0_1, upper_bound_lambda_0_3, lower_bound_lambda_0_3, now=A, nocl=5, byrow=T, 3) 13 rownames(result_table_b_0) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 13 result_table_b_0 <- c("a', b', "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 14 result_table_b_1 = round(matrix(c(a, b, mean_lambda_1, upper_bound_lambda_1, lower_bound_lambda_1, al, b1, mean_lambda_1_1, upper_bound_lambda_1_3, lower_bound_lambda_1_3, now=A, nocl=5, byrow=T, 3) 14 result_table_b_1 = round(matrix(c(a, b, mean_lambda_1, upper_bound_lambda_1, lower_bound_lambda_1, al, b1, mean_lambda_1_1, upper_bound_lambda_1_2, al, b3, mean_lambda_1_1, upper_bound_lambda_1_3, now=A, nocl=5, byrow=T, 3) 14 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 15 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 16 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 17 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 18 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 19 result_table_b_1 <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 10 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 11 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 12 rownames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") 13 result_table_b_1 <- cols.table(result_table_b_1) 14 result_table_b_1 <- cols.table(result_table_b_1) 15 result_table_b_1 <- cols.table(result_table_b_1) 16 result_table_b_1 <- cols.table(result_table_b_1) 17 result_table_b_1 <- cols.table(result_table_b_1) 18 result_table_b_1 <- cols.table(result_table_b_1) 19 result_table_b_1 <- cols.table(result_table_b_1) 19 result_table_b_1 <- cols.table(result_table_b_1) 10 result_table_b_1 <- cols.table(result_table_b_1) 10 result_table_b_1 <-</pre>	999 100 101 102 103 104 105 106 107 108 109 110 112 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128	<pre>/* prior 3 a2 = 2 b2 = 2 lambda_0.MC_2 <- rgamma(5, Y0+a2, N0+b2) lambda_1_MC_2 <- rgamma(5, Y1+a2, N1+b2) mean_lambda_0.2 = (Y0+a2)/(N0+b2) upper_bound_lambda_0.2 = ggamma(0, 025, Y0+a2, N0+b2) lower_bound_lambda_1.2 = ggamma(0, 025, Y1+a2, N1+b2) lower_bound_lambda_1.2 = ggamma(0, 025, Y1+a2, N1+b2) lower_bound_lambda_1.2 = ggamma(0, 025, Y1+a2, N1+b2) lower_bound_lambda_1.2 = ggamma(0, 025, Y1+a2, N1+b2) /* prior 4 a3 = 5 b3 = 5 lambda_0.MC_3 <- rgamma(5, Y0+a3, N0+b3) lambda_1_MC_3 <- rgamma(5, Y1+a3, N1+b3) mean_lambda_0.3 = (Y0+a3)/(N0+b3) upper_bound_lambda_0.3 = ggamma(0, 025, Y0+a3, N0+b3) lower_bound_lambda_0.3 = ggamma(0, 025, Y0+a3, N0+b3) lower_bound_lambda_0.3 = ggamma(0, 025, Y0+a3, N0+b3) mean_lambda_1.3 = (Y1+a3)/(N1+b3) upper_bound_lambda_1.3 = ggamma(0, 025, Y1+a3, N1+b3) lower_bound_lambda_1.3 = ggamma(0, 025, Y1+a3, N1+b3) /* Show results in a tables result_table_b = round(matrix(c(a,b,100*mean(lambda_1_MC_3)lambda_0_MC), a1, b1,100*mean(lambda_1_MC_1)lambda_0_MC_1), a2, b2,100*mean(lambda_0_MC_3), b7, b7, b7, b7, b7, b7, b7, b7, b7, b7</pre>
<pre>result_table_b result_table_b result_table_b.0 = round(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0,lower_bound_lambda_0,a1,b1,mean_lambda_0_1,upper_bound_lambda_0_1 ,lower_bound_lambda_0_3, nrow=4, ncol=5, byrow=7),3) rownames(result_table_b.0) <- c("Prior_1", "Prior_2", "Prior_3", "Prior_4") colnames(result_table_b.0) <- c("a", "b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") result_table_b.0 <- c("a., "b", "Mean", "Quantile (2.5%)", "Quantile (2.3%)", "Quantile (3.3, b3, mean_lambda_1_1, upper_bound_lambda_1_1 ,lower_bound_lambda_1_1, a2, b2, mean_lambda_1_2, upper_bound_lambda_1, lower_bound_lambda_1_3, now=4, ncol=5, byrow=7), 3) result_table_b.0 <- c("a", "b", "Mean", "Quantile (2.5%)", "Quantile (2.3%)", "Quantile (2.3%), mean_lambda_1_1, upper_bound_lambda_1_1, ipper_bound_lambda_1_2, ipper_bound_lambda_1_2, ipper_bound_lambda_1_2, ipper_bound_lambda_1_2, ipper_bound_lambda_1_3, ipper_bound_lambda_1_3</pre>	999 100 101 101 102 103 104 105 106 1107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 123 124 125 126 127 128 129 120	<pre>/* Prior 3 a2 = 2 b2 = 2 lambda_0.MC_2 <- rgamma(5, Y0+a2, N0+b2) lambda_1_MC_2 <- rgamma(5, Y1+a2, N1+b2) mean_lambda_0.2 = (ggamma(0, 975, Y0+a2, N0+b2) lower_bound_lambda_1.2 = (ggamma(0, 975, Y1+a2, N1+b2) lower_bound_lambda_1.2 = (ggamma(0, 975, Y1+a2, N1+b2) lower_bound_lambda_0.3 = (Y0+a3)/(N0+b3) lambda_1_MC_3 <- rgamma(5, Y1+a3, N1+b3) mean_lambda_0.3 = (ggamma(0, 975, Y0+a3, N0+b3) lower_bound_lambda_0.3 = (ggamma(0, 975, Y0+a3, N0+b3)) lower_bound_lambda_0.3 = (ggamma(0, 975, Y0+a3, N0+b3)) lower_bound_lambda_1.3 = (Y1+a3)/(N1+b3) lower_bound_lambda_1.3 = (ggamma(0, 975, Y1+a3, N1+b3)) lower_bound_lambda_1.3 = (ggamma(0, 975, Y1+a3, Y1+b3, Y1+b3,</pre>
<pre>134 135 result_table_b_0 = round(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0,lambda_0,a1,b1,mean_lambda_0_1,upper_bound_lambda_0_1 , lower_bound_lambda_0_3, norw=4, ncol=5, byrow=1),3 10 rownames(result_table_b_0) <- c(?prior1_", "Prior_2", "Prior_3", "Prior_4") 137 colnames(result_table_b_0) <- c(?a", b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 138 result_table_b_0 <- c(.a., ncol=5, byrow=1),a0 140 141 result_table_b_1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1,upper_bound_lambda_1,</pre>	9999100 1001101102 103104 105107 108107 1108107 1108107 11081107 1108 1107108 1107108 1107108 1107108 1107108 1107108 11107108 11107108 11201121 11201	<pre>// Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(5, Y0+a2, N0+b2) lambda_0_MC_2 <- rgamma(5, Y1+a2, N1+b2) mean_lambda_0_2 = (gqamma(0.975, Y0+a2, N0+b2) jower_bound_lambda_0_2 = (gqamma(0.975, Y1+a2, N1+b2) lower_bound_lambda_1_2 = qgamma(0.975, Y1+a2, N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2, N1+b2) lower_bound_lambda_1_2 = qgamma(0.025, Y1+a2, N1+b2) lower_bound_lambda_0_3 = (Y1+a3), (N0+b3) lambda_0_MC_3 <- rgamma(5, Y0+a3, N0+b3) lambda_0_3 = (Y0+a3), (N0+b3) upper_bound_lambda_0_3 = qgamma(0.025, Y0+a3, N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3, N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y0+a3, N0+b3) lower_bound_lambda_0_3 = qgamma(0.025, Y1+a3, N1+b3) lower_bound_lambda_1_3 = qgamma(0.025, Y1+a3, N1+b3) lower_bound_lambda_1_3 = qgamma(0.025, Y1+a3, N1+b3) lower_bound_lambda_1_3 = qgamma(0.025, Y1+a3, N1+b3) lower_bound_lambda_0_4 = qgamma(0.025, Y1+a3, N1+b3) lower_bound_lambda_0_4 = (P1+a3), (N1+b3) upper_bound_lambda_0_4 = (P1+a3), (N1+b3) lower_bound_lambda_0_4 = (P1+a3), (P1+a3), (P1+a3), (P1+a3), (P1+a3), (P1+a3), (P1+a3),</pre>
<pre>result_table_b_1 = round(matrix(c(a,b,mean_lambda_0,upper_bound_lambda_0,lower_bound_lambda_0,al,b1,mean_lambda_0,a,upper_bound_lambda_0,a) ,lower_bound_lambda_0_3, nrow=4, ncol=5, byrow=7),3) rownames(result_table_b_0) << c("a", "b","brior_2","Prior_3","Prior_4") 137 colnames(result_table_b_0) << c("a", "b","sman","quantile (2.5%)","quantile (97.5%)") 138 result_table_b_0 <- as.table(result_table_b_0) 140 result_table_b_0 141 result_table_b_1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1_1,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1,lower_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1_2,lower_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1, a2,b2,mean_lambda_1_2,upper_bound_lambda_1_2,lower_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 ,lower_bound_lambda_1_1, a2,b2,mean_lambda_1, "Prior_2", "Prior_3", "Prior_4") 142 colnames(result_table_b_1) <- c("Prior_1", "Prior_2", "Prior_3", "Quantile (97.5%)") 144 result_table_b_1 <- c("a", "b", "Mean", "Quantile (2.5%)", "Quantile (97.5%)") 145 result_table_b_1</pre>	9999100 1001101102 103104 105107 106107 107108 1107108 11071118 11071118 11071118 11071118 11071118 11101111111111	<pre>// prior 3 a2 = 2 b2 = 2 lambda_D_MC_2 <- rgamma(S, V)+a2, N0+b2) lambda_D_MC_2 <- (rgamma(S, V)+a2, N0+b2) lower_bound_lambda_O_2 = (qgamma(0.025, V)+a2, N0+b2) lower_bound_lambda_L_2 = (qgamma(0.025, V)+a2, N0+b2) mean_lambda_L_2 = (qyamma(0.025, V)+a2, N0+b2) lower_bound_lambda_L_2 = qgamma(0.025, V)+a2, N1+b2) lower_bound_lambda_L_2 = qgamma(0.025, V)+a2, N1+b2) lower_bound_lambda_L_2 = qgamma(0.025, V)+a2, N1+b2) lower_bound_lambda_L_3 = qgamma(0.025, V)+a2, N1+b2) lambda_D_MC_3 <- rgamma(S, V)+a3, N0+b3) lambda_D_MC_3 <- (rgamma(S, V)+a3, N0+b3) lower_bound_lambda_D_3 = (V)+a3)/(N0+b3) upper_bound_lambda_D_3 = (qgamma(0.025, V)+a3, N0+b3) lower_bound_lambda_L_3 = (qgamma(0.025, V)+a3, N0+b3) mean_lambda_L_3 = (V)+a3)/(N1+b3) upper_bound_lambda_L_3 = (qgamma(0.025, V)+a3, N0+b3) lower_bound_lambda_L_3 = qgamma(0.025, V)+a3, N0+b3) mean_lambda_L_3 = (V)+a3)/(N1+b3) upper_bound_lambda_L_3 = qgamma(0.025, V)+a3, N1+b3) /* show results in a tables result_table_b <- (C)+io_T_1, Prior_2, Prior_3, Prior_4) colnames(result_table_b) <- (C)+io_T_1, Prior_2, Prior_4, Prior_4) result_table_b <- (C)+io_T_1, Prior_2, Prior_4, Prior_4, Prior_4) result_table_b <- (C)+io_T_1, Prior_2, Prior_4, Prior_4) result_table_b, <- (C)+io_T</pre>
<pre>, lower_bound_lambda_0_3), nrow=4, ncol=5, byrow=1),3) rownames(result_table_b_0) <- c("prior_1", "prior_2", "prior_3", "prior_4") rownames(result_table_b_0) <- c("ar,"b,",wean", quantile (2.5%)", "quantile (97.5%)") result_table_b_0 <- as.table(result_table_b_0) result_table_b_1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1_1,upper_bound_lambda_1_1,</pre>	99991000 10011012103 1004105 1006107108 1106107108 110911011111111111111111111111111111	<pre>// prior 3 a2 = 2 b2 = 2 lambda_0_K_2 <~ rgamma(5, Yl+a2, Nl+b2) mean_lambda_0_2 = (y0+a2)/(Nl+b2) upper_bound_lambda_0_2 = ggamma(0.975, Yl+a2, Nl+b2) lower_bound_lambda_1_2 = (ggamma(0.925, Yl+a2, Nl+b2) lower_bound_lambda_1_2 = ggamma(0.975, Yl+a2, Nl+b2) lower_bound_lambda_1_2 = ggamma(0.975, Yl+a2, Nl+b2) // prior 4 a3 = 5 b3 = 5 lambda_0_KC_3 <~ rgamma(5, Yl+a3, Nl+b3) mean_lambda_0_3 = (y0+a3)/(Nl+b3) upper_bound_lambda_0_3 = ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_1_3 = (ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_1_3 = ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_0_3 = ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_0_3 = ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_0_3 = (ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_0_3 = (ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_1_3 = (ggamma(0.975, Yl+a3, Nl+b3) lower_bound_lambda_0_0_C < (Prior_1, YPrior_2, Yl+a3, Nl+b3) // show result table_b << ((Prior_1, YPrior_2, YPrior_3, YPrior_3, YPrior_4, YPrior_3, YPrior_4, YPrior_3, YPrior_4, YPrior_3, YPrior_4, YPrior_3, YPrior_4, YPrior_3, YPrior_4, YPrior_4, YPrior_3, YPrior_4, Y</pre>
<pre>130 rownames(result_table_0) <- c("prior_1, "prior_2, prior_3, prior_4) 131 colnames(result_table_0) <- c("ar,"b,","kean,", quantile (2.5%)", "quantile (97.5%)") 132 result_table_0 <- as.table(result_table_0) 133 result_table_0</pre>	9999100 1001101102 1103104 1105106 1106107 1108109 11001111 1122113 11141115 1116117 1122123 1226 1227 1229 13001221 132129 130013134 1331134	<pre>/* Prior 3 a2 = 2 b2 = 2 lambda_0_MC_2 <- rgamma(S,Y0+a2,N0+b2) lambda_0_Z = (Y0+a2)/(N0+b2) upper_bound_lambda_0_Z = ggamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_1_Z = ggamma(0.975, Y0+a2,N0+b2) lower_bound_lambda_1_Z = ggamma(0.925, Y0+a2,N0+b2) lower_bound_lambda_1_Z = ggamma(0.975, Y1+a2,N1+b2) lower_bound_lambda_1_Z = ggamma(0.975, Y1+a2,N1+b2) lower_bound_lambda_1_Z = ggamma(0.975, Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(S,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(S,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(S,Y0+a3,N0+b3) lambda_1_ambda_0_3 = ggamma(0.975, Y0+a3,N0+b3) lower_bound_lambda_1_3 = ggamma(0.975, Y1+a3,N1+b3) lower_bound_lambda_1_3 = ggamma(0.975, Y1+a3,N1+b3) lower_bound_lambda_1_3 = ggamma(0.975, Y1+a3,N1+b3) lower_bound_lambda_1_3 = ggamma(0.975, Y1+a3,N1+b3) lower_bound_lambda_0_MC_3, Statables result_table_b = round(matrix(c(a,b,100*mean(lambda_1_MC_3)ambda_0_MC_3), a1,b1,100*mean(lambda_1_MC_1>lambda_0_MC_1),a2,b2,100*mean (lambda_1_MC_2).aabbd_0_MC_2), a3,b3,100*mean(lambda_1_MC_3)ambda_0_MC_3), nrow=4, ncol=3, byrow=1),3) rownames(result_table_b) <- c(Fungt_*, "Prior_3", "Prior_4") result_table_b <- cound(matrix(c(a,b, 0_0, upper_bound_lambda_0_N(a, 0, lower_bound_lambda_0, a1, b1, mean_lambda_0_1, upper_bound_lambda_0_3 result_table_b <- cound(matrix(c(a,b, upor_bound_lambda_0, 0, lower_bound_lambda_0, a1, b1, mean_lambda_0_1, upper_bound_lambda_0_3 result_table_b <- cound(matrix(c(a,b, upor_bound_lambda_0, 0, lower_bound_lambda_0, a1, b1, mean_lambda_0_1, upper_bound_lambda_0_3</pre>
<pre>138 result_table_b_0 <- as.table(result_table_b_0) 139 result_table_b_0 140 141 result_table_b_1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1_1,upper_bound_lambda_1_1 , lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1_2,lower_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 , lower_bound_lambda_1_3, nrow=4, ncol=5, byrow=7),3) 142 rownames(result_table_b_1) <- c("prior_1","Prior_2","Prior_3","Prior_4") 143 colnames(result_table_b_1) <- c("ar,"b","Mean","Quantile (2.5%)","Quantile (97.5%)") 144 result_table_b_1 <- as.table(result_table_b_1) 145 result_table_b_1</pre>	99991 1001101102 103104 105106 107108 1091107 1108 110911107 1112111111111111111111111111111111	<pre>// Prior 3 a2 = 2 b2 = 2 babda_LMC_2 <- rgamma(5,Y0+a2,N0+b2) lambda_LMC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = (y0+a2)/(y0+b2) upper_bound_lambda_0_2 = ggamma(0.025, Y0+a2,N0+b2) lower_bound_lambda_0_2 = ggamma(0.025, Y0+a2,N0+b2) lower_bound_lambda_1_2 = (ggamma(0.025, Y1+a2,N1+b2) lower_bound_lambda_1_2 = ggamma(0.025, Y1+a2,N1+b2) // Prior 4 a3 = 5 b3 = 5 b3 = 5 b3 = 5 lambda_0_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_LMC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- rgamma(5,Y0+a3,N0+b3) lambda_1_MC_3 <- (rgamma(5,Y0+a3,N0+b3)) lower_bound_lambda_1_3 = (y1+a3)/(N1+b3) upper_bound_lambda_1_3 = (y1+a3)/(N1+b3) lower_bound_lambda_1_3 = (y1+a3)/(N1+b3) lower_bound_lambda_1_3 = (y1+a3)/(N1+b3) upper_bound_lambda_1_3 = (y1+a3)/(N1+b3) lower_bound_lambda_1_3 = (y1+a3)/(N1+b3) lower_bound_lambda_0_0, wc_2), a; b; 100*mean(lambda_1_Wc_3)lambda_0_Mc_3)), nrow=4, ncol=3, byrow=1), a; b; 2,100*mean(lambda_0_1, a; b; 2,100*mean(lambda_0_2, a; b; 3, 100*mean(lambda_0_2, lambda_0_0, lower_bound_lambda_0_3, lower_bound_lambda_0_3,</pre>
<pre>139 result_table_b_0 140 141 result_table_b_1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1_1,upper_bound_lambda_1_1 , lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1_2,lower_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3 , lower_bound_lambda_1_3, nrow=4, ncol=5, byrow=7,3) 142 rownames(result_table_b_1) <- c("prior_1", "prior_2", "prior_3", "prior_4") 143 colnames(result_table_b_1) <- c("a", "b", "mean", "quantile (2.5%)", "quantile (97.5%)") 144 result_table_b_1 <- as.table(result_table_b_1) 145 result_table_b_1</pre>	99991 10011012 103104 105106 107108 1091107 1108 1091107 1112 11131114 1115 1116 1117 1118 1116 1117 1122 123127 128 129 130 131 133 134 135 136 137	<pre>/* Prior 3 a2 = 2 b2 = 2</pre>
<pre>141 result_table_b_1 = round(matrix(c(a,b,mean_lambda_1,upper_bound_lambda_1,lower_bound_lambda_1,a1,b1,mean_lambda_1_1,upper_bound_lambda_1_1,</pre>	99991 1001101 102103 104105 1061107 108109 1107108 109111111111111111111111111111111111	<pre># Prior 3 a2 = 1 a12 = (y1+2)/(y1+2) upper_bound_lambda_0_2 = (gamma(0,075, y0+2,N0+b2) lower_bound_lambda_0_2 = (gamma(0,075, y0+2,N0+b2) lower_bound_lambda_0_3 = (y0+23)/(N0+b3) lambda_0_NC_3 < rgamma(5,y0+3,N0+b3) lambda_0_NC_3 < rgamma(5,y0+3,N0+b3) lambda_0_NC_3 < rgamma(5,y0+3,N0+b3) lambda_0_3 = (y0+23)/(N0+b3) upper_bound_lambda_0_3 = (gamma(0,075, y0+3,N0+b3) lower_bound_lambda_0_3 = (gamma(0,075, y0+3,N0+b3)) lower_bound_lambda_0_3 = (ga</pre>
<pre>, lower_bound_lambda_1_1,a2,b2,mean_lambda_1_2,upper_bound_lambda_1_2,lower_bound_lambda_1_2,a3,b3,mean_lambda_1_3,upper_bound_lambda_1_3</pre>	99901101 1001101102 10031004 10051006 10071008 10070000000000	<pre>/ Prior 3 a2 = 2 b2 = 2 lambda_D.K.2 <- (rgamma(5,Y0+a2,N0+b2) lambda_LJKC.2 <- (rgamma(5,Y1+a2,N1+b2) mean_lambda_D.2 = ggamma(0:05; Y0+a2,N0+b2) lower_bound_lambda_D.2 = ggamma(0:05; Y0+a2,N0+b2) lower_bound_lambda_D.2 = ggamma(0:05; Y1+a2,N1+b2) lower_bound_lambda_L1 = qgamma(0:02; Y1+a2,N1+b2) lower_bound_lambda_L1 = qgamma(0:02; Y1+a2,N1+b2) lower_bound_lambda_L2 = (rgamma(0:02; Y1+a2,N1+b2) lower_bound_lambda_L3 = (rgamma(0:02; Y1+a2,N1+b2) lower_bound_lambda_L3 = (rgamma(0:02; Y1+a2,N1+b2) lower_bound_lambda_L3 = (rgamma(0:02; Y1+a3,N1+b3) mean_lambda_L3 = (rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = (rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = qgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = (rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = qgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = qgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = (rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = (rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3 = rgamma(0:02; Y1+a3,N1+b3) lower_bound_lambda_L3, rgamma(0:02; Y1+a3,N1+b3) result_table_b - <- connd(matrix(c(a,b,mean_lambda_0,MC_3), rrow=, ncol=; byrow=),;) rownames(result_table_b), rrow=, rool=, byrow=),;) rownames(result_table_b), rrow=, rool=, byrow=),;) rownames(result_table_b, rool=, rool=, byrow=),;) rown</pre>
, Tower_bound_lambda_l_3), nrow=4, ncol=>, byrow=1),3) 142 rownames(result_table_b_1) <- c("prior_1","Prior_3","Prior_4") 143 colnames(result_table_b_1) <- c("a","b","Mean","Quantile (2.5%)","Quantile (97.5%)") 144 result_table_b_1 <- as.table(result_table_b_1) 145 result_table_b_1 146	99901101 10011012 1001101103 11041105 1107108 1100111111111111111111111111111	<pre>/* Prior 3 a2 = 1 abdd_0_MC_2 <- rgamma(5,Y0+a2,N0+b2) lambdd_1_MC_2 <- rgamma(5,Y1+a2,N1+b2) mean_lambda_0_2 = ggamma(0.025,Y0+a2,N0+b2) lower_bound_lambda_0_2 = ggamma(0.025,Y0+a2,N0+b2) lower_bound_lambda_0_2 = ggamma(0.025,Y1+a2,N1+b2) lower_bound_lambda_1_2 = ggamma(0.025,Y1+a2,N1+b2) lower_bound_lambda_1_2 = ggamma(0.025,Y1+a2,N1+b2) lower_bound_lambda_0_3 = (Y1+a3)/(N1+b3) mean_lambda_0_MC_3 <- rgamma(0.025,Y1+a2,N1+b2) lower_bound_lambda_0_3 = ggamma(0.025,Y1+a2,N1+b2) lower_bound_lambda_0_3 = ggamma(0.025,Y1+a3,N1+b3) mean_lambda_1_MC_3 <- rgamma(0.975,Y0+a3,N0+b3) lower_bound_lambda_0_3 = ggamma(0.025,Y1+a3,N1+b3) mean_lambda_1_MC_3 <- rgamma(0.975,Y0+a3,N0+b3) lower_bound_lambda_0_3 = ggamma(0.025,Y1+a3,N1+b3) mean_lambda_1_3 = (Y1+a3)/(N1+b3) upper_bound_lambda_0_3 = ggamma(0.025,Y1+a3,N1+b3) mean_lambda_1_3 = (Y1+a3)/(N1+b3) upper_bound_lambda_0_3,Y1+a3,N1+b3) mean_lambda_1_3 = (Y1+a3)/(N1+b3) mean_lambda_1_3 = (Y1+a3)/(N1+b3) mean_lambda_1_3</pre>
143 colnames(result_table_b_1) <- c("a","b","hean","Quantile (2.5%)","Quantile (97.5%)") 144 result_table_b_1 <- as.table(result_table_b_1) 145 result_table_b_1 146	$\begin{array}{c} 999\\ 9101\\ 1001\\ 1002\\ 1001\\ 1003\\ 1004\\ 1005\\ 1007\\ 1008\\ 1007\\ 1008\\ 1007\\ 1008\\ 1010\\ 1111\\ 1110\\ 1110\\ 1111\\ 1110\\ 1110\\ 1111\\ 1110\\ 1100\\ $	<pre>/ Prior 3 p2 = 2 p2 = 2 lambda_D.W.2 <- (rgamma(5, Y0+32, N0+b2) lambda_J_W.2 <- (rgamma(5, Y0+32, N0+b2) lambda_J_W.2 <- (rgamma(5, Y0+32, N0+b2) lower_bound_lambda_D.2 = ggamma(0.035, Y0+32, N0+b2) lower_bound_lambda_L =- (Y1+31) (VM+b1) upper_bound_lambda_L = - (Y1+31) (VM+b2) // Prior 4 d3 = 5 b3 = 5 lambda_D.W.3 <- rgamma(5, Y0+32, N0+b3) lambda_D.W.3 <- rgamma(5, Y0+33, N0+b3) lambda_D.S.3 <- rgamma(5, Y0+33, N0+b3) lower_bound_lambda_L3 = (Q0+33) (VM+33) upper_bound_lambda_D.3 = ggamma(0.025, Y0+33, N0+b3) lower_bound_lambda_D.3 = ggamma(0.025, Y0+33, N0+b3) lower_bound_lambda_D.3 = ggamma(0.025, Y1+33, N1+b3) lower_bound_lambda_D.3 = ggamma(0.025, Y1+33, Y1+b3) lower_bound_lambda_D.3 = ggamma(0.025, Y1+33, Y1+b3) lower_bound_lambda_D.3 = ggamma(0, 025, Y1+33, Y1+b3) lower_bound_lambda_D.3 = ggamma(0</pre>
144 result_table_b_1 <- as.table(result_table_b_1) 145 result_table_b_1 146	9991101 1001100 11011103 11041105 1107108 1107108 111071108 111071108 111071108 111071108 111071108 111071108 111071108 11201110 11201110 1131113 1136 11371138 11381139 11401 1141113 11381139 11401 1141113 11381139 11401 1141113 11381139 11401 1141113 11381139 11401 1141113 11381139 11401 1141113 11381113 113811111111	<pre>/ Prior 3 22 = 2 1anbda_UKC2 <- rgamma(5, V0+a2, N0+b2) 1anbda_UKC2 <- rgamma(5, V0+a2, N0+b2) 1anbda_UKC2 <- rgamma(5, V0+a2, N0+b2) 1ower_bound_lambda_UC2 = qgamma(0, 055, V0+a2, N0+b2) 1ower_bound_lambda_UC2 = qgamma(0, 055, V0+a2, N0+b2) 1ower_bound_lambda_UC2 = qgamma(0, 055, V1+a2, N1+b2) 1ower_bound_lambda_UC2 = qgamma(0, 055, V1+a3, N1+b3) 1ambda_UKC3 <- rgamma(5, V0+a3, N0+b3) 1ambda_UKC3 <- rgamma(5, V0+a3, N0+b3) 1ambda_UKC3 <- rgamma(5, V0+a3, N0+b3) 1ower_bound_lambda_UC3 = qgamma(0, 057, V1+a3, N1+b3) 1ower_bound_lambda_UC3 = qgamma(0, 057, V1+a3, N1+b3) * show results in a tables result_table_b - round(matris(C(a, b, 100*mean(lambda_UKC3) = result_table_b, C(a, b, 100*mean(lambda_UKC3), rrom=(a, ncol=3, byrom=1), 3) result_table_b - cound(matris(C(a, b, gean_lambda_0, ypper_bound_lambda_0, A, lob, mean(lambda_0, L, C) lambda_0, U, lob, mean(lambda_0, L, C) lambda_0, L, lob, mean(lambda_0, L, L), lambda_0, L, lob, mean(lambda_0, L, L), lambda_0, L, lob, mean(lambda_0, L, VD+Dound_lambda_0, L, lob, mean(lambda_0, L, VD+Dound_lambda_0, L, L), lambda_0, L, lob, mean(lambda_0, L, VD+Dound_lambda_0, L, L), lambda_0, L, lob, mean(lambda_0, L, VD+Dound_lambda_0, L, lob, mean(lambda_0, L, VD+Dound_lambda_0, L, L), lob, mean(lambda_0, L, VD+Dound_lambda_0, L, lob, mean(lambda_0, L), lob, mean(lambda_0, L, L), lob, mean(lambda_0, L), mean(lambda_0, L), lob, mean(lambda_0, L), lob, mean(lambda_0, L), lob, mean(lambda_0, L), mean(lambda_0, L), lob, mean(lambda_0, L), lob, mean(lambda_0, L), m</pre>
	$\begin{array}{rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr$	<pre># // // // // // // // // // // // // //</pre>
	$\begin{array}{rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr$	<pre># /* / / / / / / / / / / / / / / / / / /</pre>

147 -	нини	
147 ¥ 148		
149 -	##### # Load known data	
151	NO_France = 398	
152 153	Y0_France = 33 a = 0.10	
154	b = 0.10	
155 156		
157	# Plot the fit with data points of other countries and 95% prediction intervals	
158	y = medals\$PARTICIPATING.ATHLETES.DURING.PREVIOUS.OLTMPICS	
160	fit <- lm(y~x)	
162	<pre>summary(rec) # recsummary samples <- seq(min(x),max(x),1)</pre>	
163	Host_Participants = fit\$coefficients[1] + fit\$coefficients[2]*samples	-'nrediction')
165	<pre>pred_interval = predict(fit,data.frame(x=c(samples)),level=0.95,interval='predict</pre>	tion')
166	<pre>plot(x,y, pch=19,xlab="Participants of Non-Host Country", ylab="Participants of (Host Participants)))</pre>	Host Country", xlim=c(min(x),max(x)),ylim=c(min(y),max
167	<pre>lines(samples,Host_Participants,col='blue',lwd=3)</pre>	
168 169	lines(samples,pred_interval[,2],col= green ,lwd=3) lines(samples,pred_interval[,3],col='green',lwd=3)	
170	<pre>legend("topleft",legend=c("N1 = 216.068 + 0.867*N0","95% PI","actual"),lwd=c(3,3 "black"))</pre>	<pre>,NA),pch=c(NA,NA,19),lty=c(1,1,NA),col=c("blue","green"</pre>
171	, DIACK))	
172	<pre># Predict Y1_France, i.e. the number of medals won in 2024 from France, using Lk x modals = modalsSurpaus won puptic provides of ymptrs</pre>	
174	y_medals = medals\$MEDALS.WON.DURING.HOST.YEAR	
175 176	<pre>fit_medals <- lm(y_medals~x_medals) summary(fit_medals) # fit_summary</pre>	
177	<pre>samples_medals <- seq(min(x_medals), max(x_medals), 1)</pre>	
178 179	<pre>Host_medals = fit_medals\$coefficients[1] + fit_medals\$coefficients[2]*samples_met plot(x medals.v medals. pch=19.xlab="Medals of Non-Host Country".vlab="Medals c</pre>	dals wf Host Country". xlim=c(min(x medals).max(x medals)).vlim=c
100	(min(y_medals),max(Host_medals)))	
180	<pre>lines(samples_medals, Host_medals, Cole blue , Host3) legend("topleft", legend=c("Y1 = 4.319 + 1.376*Y0", "actual"), lwd=c(3, NA), pch=c(NA</pre>	,19),lty=c(1,NA),col=c("blue","black"))
182		
184	S = 100000	y statistics are computed
185 186	set.seed(1) Y star w unc = ren(0.5)	
187	index = 1	
188 - 189	<pre>tor (1 1n 1:5){ lambda_star = roamma(1.Y0_France+a.N0_France+b)</pre>	
190	N1_France_unc = rnorm(1,pred_interval_N0_France[,1],(pred_interval_N0_France[,	3]-pred_interval_N0_France[,1])/2)
191 192	<pre>index = index + 1</pre>	
193 •	} hist(Vistar willno)	
195	Y_star_w_unc_mean = mean(Y_star_w_unc)	
	\mathbf{V} star w ups sd = sd(\mathbf{V} star w ups)	
196 197	Y_star_w_unc_2_5_quantile = quantile(Y_star_w_unc,0.025)	
196 197 198	Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_97_5_quantile = quantile(Y_star_w_unc,0.975)	
196 197 198	Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975)	
196 197 198 199 200	<pre>/</pre>	ummary statistics are computed
196 197 198 199 200 201 202	<pre>%_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(5, Y0_France+a, N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France</pre>	ummary statistics are computed
196 197 198 199 200 201 202 203	<pre>%_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(s, Y0_France+a, N0_France+b) NL_France = fits(coefficients[1] + fits(coefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star)</pre>	ummary statistics are computed
190 197 198 200 201 202 203 204 205	<pre>%_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(s, Y0_France+a, N0_France+b) NL_France = fitsCoefficients[1] + fitsCoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) </pre>	ummary statistics are computed
196 197 198 199 200 201 202 203 204 205 206 207	<pre>%_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(s, Y0_France+a, N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star)</pre>	ummary statistics are computed
199 197 198 200 201 202 203 204 205 206 207 208	<pre>%_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(s, Y0_France+a, N0_France+b) NL_France = fit\$coefficients[1] + fit\$coefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_sd = sd(Y_star) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_25_quantile = quantile(Y_star,0.975)</pre>	ummary statistics are computed
190 197 198 200 201 202 203 204 205 206 207 208 209 210	<pre>%_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.075) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(s, Y0_France+a, N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_9_5_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star,0.025) # Plot with and without NI_France uncertainty in PPD</pre>	ummary statistics are computed
199 197 198 200 201 202 203 204 205 206 207 208 209 210 211	<pre>%_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(s, Y0_France+a, N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_2.5_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star,0.025) # Plot with and without N1_France uncertainty in PPD plot(NULL, xlim=c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals Won")</pre>	ummary statistics are computed
199 197 198 200 201 202 203 204 205 206 207 208 209 210 211 212 213	<pre>Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_sda = sd(Y_star) Y_star_sda = sd(Y_star) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star,0.025) Plot(with and without NL_France uncertainty in PPD Plot(with, xlim=c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_NL_unc = density(Y_star_w_unc) </pre>	ummary statistics are computed
1997 1998 1997 2000 2011 2022 203 204 205 206 207 208 209 210 211 212 213 214 215	<pre>Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) # Plot with and without NL_France uncertainty in PPD plot(NULL, xlim=c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col="red",lwd=3) lines(dens_w_NL_uncY_data)*dens_w_NL_uncSy,type="1",col="red",lwd=3) lines(dens_w_NL_uncY_data)*dens_w_NL_uncSy,type="1",col="red",lwd=3) lines(dens_w_NL_uncY_data)*dens_w_NL_uncYY_type="1",col="red",lwd=3) lines(dens_w_NL_uncY_data)*dens_w_NL_uncYY_type="1",col="red",lwd=3) lines(dens_w_NL_uncY_data)*dens_w_NL_uncYY_type="1",col="red",lwd=3) lines(dens_w_NL_uncY_data)*dens_w_NL_uncYY_type="1",col="red",lwd=3) lines(dens_w_NL_uncY_data)*dens_w_NL_uncYY_type="1",col="red",lwd=3) lines(dens_w_NL_uncYY_type="1",col="red",lwd=3) lines(d</pre>	ummary statistics are computed
197 197 198 199 200 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216	<pre>Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Plot(NuLL, xlim=c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_N1_unc = density(Y_star_w_unc) dens_w_N1_unc = density(Y_star) lines(dens_w_N1_uncSx,length(data)*dens_w_N1_uncSy,type="1",col="red',lwd=3) lines(dens_w_N1_uncSx,length(data)*dens_w_N1_uncSy,type="1",col="red',lwd=3) legend("topright",legend-c('with N1_France Unc', without N1_France Unc'), pch= </pre>	ummary statistics are computed rep(0,2),1wd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1
190 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217	<pre>Y_star_w_unc_2_Squartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_2_Squartile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_de = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Plot(Nutl, xlim=(c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_N1_unc = density(Y_star_w_unc) dens_w_N1_unc = density(Y_star) lines(dens_w_N1_unc\$x,length(data)*dens_w_N1_unc\$y,type="1",col='red',lwd=3) legend("topright",legend=c('with N1_France Unc', 'without N1_France unc'), pch = }</pre>	ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1
1907 1977 198 1997 2000 2012 2002 2003 2004 2005 2005 2007 2007 2007 2007 2007 2102 2112 2112	<pre>Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_star = rpois(S,NL_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_de = sd(Y_star) Y_star_sd = sd(Y_star) Y_star_sd = sd(Y_star) Plot(With and without NL_France uncertainty in PPD Plot(With, xlim=(c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_uncSx,l</pre>	ummary statistics are computed rep(0,2),1wd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1
1907 1977 198 2000 2012 2022 2032 2042 2052 2062 2072 2082 2072 2082 2072 2082 2092 2102 2112 2122 2113 2142 2152 216 2172 2182 219	<pre>Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,YO_France+a,NO_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*NO_France Y_star = rpois(S,NL_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_sd = sd(Y_star) Ines(dens_w_NL_unc`s,length(data)*dens_w_NL_unc`sy,type="1",col='red',lwd=3) lines(dens_w_NL_unc`s,length(data)*dens_w_NL_unc`sy,type="1",col='fue',lwd=3) liegend("topright",length(data)*dens_w_NL_unc`sy,type="1",col='fue',lwd=3) Isegure("topright",length(data)*dens_w_NL_unc`sy,type="1",col='fue',lwd=3) Y_star_sd = sd(Y_star_N_w_unc) # show results in a table result_table_PPD = matrix(c(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2.5_g </pre>	ummary statistics are computed rep(0,2),1wd=rep(2,2),co1=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd
1907 1977 198 1997 2000 2011 2022 2022 2022 2022 2023 2024 205 207 208 209 210 211 212 213 214 215 215 216 217 218 219 220 221	<pre>Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_star = rpois(S,NL_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_sd = sd(Y_star) Hold with and without NL_France uncertainty in PPD plot(NuL, xlim=(c(0,120), ylim=c(0,005), ylab="PPD", xlab="Medals won") dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_uncSy,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_uncSy,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_uncSy,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd</pre>	ummary statistics are computed rep(0,2),1wd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd Uncertainty'')
1907 1977 198 1997 2001 2012 203 204 205 206 207 208 209 210 211 212 212 213 214 215 216 217 2218 219 220 221 2221 2222	<pre>Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, NC sampling is performed and s lambda_star = rgamma(5,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_star = rpois(S,NL_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_star_sd = sd(Y_star) Y_star_25_guantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star_w_unc) dens_w_N1_unc = density(Y_star) lines(dens_w_N1_unc Sx, length(data)*dens_w_N1_uncSy,type="1",col='red',lwd=3) lines(dens_w_N1_uncSx,length(data)*dens_w_N1_uncSy,type="1",col='red',lwd=3) liegend("topright",legend=c(`with N1_France Unc', `without N1_France Unc'), pch =) # Show results in a table result_table_PPD = matrix(c(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2_5_quantile, ystar_95_quantile), nrow=2, ncol=4, byrow=T) rownames(result_table_PPD) <- c(`With N1_France Uncertainty', `Without N1_France colnames(result_table_PPD) <- c(`With N1_France Uncertainty', `Without N1_France colnames(result_table_PPD) <- c(`With N1_France Uncertainty', `Without N1_France colnames(result_table_PPD) <- c(`With N1_France Uncertainty', `Without N1_Fr</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd Uncertainty") ")</pre>
1907 1977 198 1997 2002 2012 2023 204 2052 206 2072 208 209 210 212 212 212 212 212 212 212 212 212	<pre>Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_27_5_quantile = quantile(Y_star_w_unc,0.075) # Not including NL_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_de = sd(Y_star) Y_star_sd = sd(Y_star) Ines(dens_w_N1_unc = density(Y_star_w_unc) dens_w_N1_unc = density(Y_star) lines(dens_w_N1_unc\$x,length(data)*dens_w_N1_unc\$y,type="1",col='red',lwd=3) lines(dens_w_N1_unc\$x,length(data)*dens_w_N1_unc\$y,type="1",col='blue',lwd=3) lines(dens_w_N1_unc\$x,length(data)*dens_w_N1_unc\$y,type="1"</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd Uncertainty") ")</pre>
190 197 197 197 197 197 197 197 200 200 200 200 203 203 204 205 206 207 208 209 211 212 213 214 215 217 217 218 219 220 221 221 221 221 221 221 221	<pre>Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_27_5_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_27_5_quantile = quantile(Y_star_w_unc,0.025) NL_France = fitscoefficients[1] + fitscoefficients[2]*NO_France NL_France = fitscoefficients[1] + fitscoefficients[2]*NO_France Y_star_star = ropsis(S,NL_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_sd = sd(Y_star) Ines(dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) Ines(dens_w_NL_unc\$x,length(data)*dens_w_NL_unc\$y,type="1",col='red',lwd=3) Ines(dens_w_NL_unc\$x,length(data)*dens_w_NL_unc\$y,type="1",col='blue',lwd=3) Ines(dens_w_NL_unc\$x,length(data)*dens_w_NL_unc\$y,type=</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd Uncertainty") ")</pre>
190 197 197 197 197 197 197 197 200 200 200 200 203 203 204 205 203 204 205 206 207 208 209 211 212 213 214 215 217 217 221 221 221 221 221 221	<pre>Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_27_5_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_27_5_quantile = quantile(Y_star_w_unc,0.025) Y_star_star = rgamma(5,Y0_France+a,N0_France+b) M_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_sd = sd(Y_star) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star,0.025) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_25_quantile = quantile(Y_star_w_unc) dens_w_N1_unc = density(Y_star) lines(dens_w_N1_unc's,1,length(data)*dens_w_N1_unc'sy,type="1",col='red',lwd=3) lines(dens_w_N1_unc's,1,length(data)*dens_w_N1_unc'sy,type="1",col='blue',lwd=3) legend("topright",legend=c(`with N1_France Unc', `without N1_France Unc'), pch =) # Show results in a table result_table_PPD = matrix(c(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2_5.quantile(2.5%)", "Quantile (97.5%)", result_table_PPD - c c(`With N1_France Uncertainty", `Without N1_France colnames(result_table_PPD) <- c(`Whan', `std', `Quantile (2.5%)', `Quantile (97.5%)", result_table_PPD ###################################</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd Uncertainty") ")</pre>
190 197 198 199 200 201 2023 204 205 207 206 207 208 209 210 211 212 213 214 215 216 221 221 221 221 221 221 221 221 221 221 2221 2223 2224 2254 2274 228	<pre>Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_star = rgamma(5,Y0_France+a,N0_France+b) M_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_mean = mean(Y_star) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star) lines(dens_w_Al_unc_5x,length(data)*dens_w_Al_unc\$y,type="1",col='red',lwd=3) lines(dens_wo_Al_unc\$x,length(data)*dens_wo_Al_unc\$y,type="1",col='blue',lwd=3) liegend("topright",legend=c(`with N1_France Unc', `without N1_France Unc'), pch =) # Show results in a table result_table_PPD = matrix(c(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2_5.0 Y_star_25_quantile(Y_star_97_5_quantile), nrow=2, ncol=4, byrow=T) rownames(result_table_PPD) <- c(`Waan', "std", "quantile (2.5%)", "quantile (97.5%) result_table_PPD ##### # Load in Data N0_per_country = c(129, 135, 94, 275, 208, 410, 175, 229, 941, 498, 140, 384, 30 </pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557)</pre>
190 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 2221 2221 2221 2221 2221 2221 2221 2223 2241 2250 2220 2221 2223 2224 2226 2230 2230 2230 2230 2231	<pre>Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPO, NC sampling is performed and s lambda_star = rgamma(5,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_w_unc_25_quantile = quantile(Y_star_N0_France+b) N_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_w_unc_25_quantile = quantile(Y_star,0.025) Y_star_mean = mean(Y_star) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star) lines(dens_wo_NL_uncsx,length(data)*dens_wo_NL_unc\$y,type="1",col='red',lwd=3) lines(dens_wo_NL_unc\$x,length(data)*dens_wo_NL_unc\$y,type="1",col='blue',lwd=3) liegend("topright",legend=c('with N1_France Unc', 'without N1_France Unc'), pch =) # Show results in a table result_table_PPD = matrix((Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2_5.quantile), nrow=2, ncol=4, byrow=T) rownames(result_table_PPD) <- c("With N1_France Uncertainty", "Without N1_France colnames(result_table_PPD <- c("With N1_France Uncertainty", "Without N1_France colnames(result_table_PPD) <- c("Wean", "std", "Quantile (2.5%)", "Quan</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557) 9, 530, 462, 949)</pre>
190 197 198 1900 2001 2002 2003 2004 2005 2006 2007 2008 2012 2112 2113 2114 2115 2116 2127 2128 2219 2221 2221 2222 2231 2230 2231 2320 2301 2321 2321	<pre>Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPD, NC sampling is performed and s lambda_star = rgamma(5,Y0_France+a,N0_France+b) NL_France = fitScoefficients[] + fitScoefficients[2]*N0_France Y_star_w_unc_25_quantile = quantile(Y_star,0.025) Y_star_star_sd = sd(Y_star) Y_star_25_quantile = quantile(Y_star,0.025) Y_star_97_5_quantile = quantile(Y_star_w_unc) dens_w_N1_unc = density(Y_star) lines(dens_w_N1_unc_5,1ength(data)*dens_w_N1_unc\$y,type="1",col='red',lwd=3) lines(dens_w_N1_unc\$x,length(data)*dens_w_N1_unc\$y,type="1",col='blue',lwd=3) liegend("topright",legend=c('with N1_France Unc', 'without N1_France Unc'), pch =) # Show results in a table result_table_PPD = matrix(C(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2_5.2,quantile(Y.5x)", "Quantile (Y.5x)", "Quantile</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd Uncertainty") 4, 236, 557) 9, 530, 462, 949)</pre>
190 197 198 199 200 2012 203 2042 205 2067 2082 2012 2112 2112 2132 214 215 216 217 218 219 2201 2212 2213 2224 2225 2230 2231 2322 2301 2322 2301 2322 231 2322 231 2322 231 232 2331	<pre>Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_quantile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not finct(Y_star) Y_star_star_star) hist(Y_star) Y_star_star_star) Y_star_star_star_guantile = quantile(Y_star,0.025) Y_star_star_star_star) plot(with, xlim=c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_uncSx,length(data)*den</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557) 9, 530, 462, 949) iet Union', 'South Korea', 'Spain', 'United States',</pre>
1907 1977 198 1997 2001 2002 2003 2002 2003 2004 2005 2007 2008 210 2102 2103 2102 2103 2102 2112 2112	<pre>Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Not finct(Y_star) Y_star_star) Y_star_star) Y_star_star_star) Y_star_star_star_star,0.025) Y_star_star_star_star_ytar_ytar_star_star_star_star_star_star_star_s</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557) 9, 530, 462, 949) iet Union', 'South Korea', 'Spain', 'United States',</pre>
1907 1977 198 1997 2001 2002 2003 2002 2003 2004 2005 2007 2008 2102 2103 2102 2103 2102 2112 2123 2112 2123 2112 2123 2114 2125 2224 2224 2225 2224 2225 2226 2227 2228 2230 2231 2232 2232 2232 2232 2232 2232	<pre>Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guartile = quantile(Y_star_w_unc,0.975) # Not including NL_France quartile (Y_star_w_unc,0.975) # Jot with and without NL_France location (Y_star,0.025) Y_star_Sd = sd(Y_star) Y_star_sd = sd(Y_star) # Plot with and without NL_France uncertainty in PPD plot(NuLl, xlim=c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_unc) = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) * fand without NL_France Unc', 'without NL_France Unc'), pch = } # Show results in a table result_table_PPD = matrix(c(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2.5.q Y_star_2.5.quantile, ystar_9.5.quantile), nrow=2, ncol=4, byrow=1) rownames(result_table_PPD) <- c("With NL_France Uncertainty", "Without NL_France colnames(result_table_PPD) <- c("Wean", "std", "Quantile (2.5%)", "Quantile (97.5%) result_table_PPD = (129, 135, 94, 275, 208, 410, 175, 229, 941, 498, 140, 384, 300 NLper_country = c(129, 135, 94, 275, 208, 410, 175, 229, 941, 498, 140, 384, 300 NLper_country = c(22, 36, 94, 011, 125, 32, 275, 93, 16, 100, 65, 19, 800 Country_names = c('Finland', 'Ttaly', 'Mexico', 'West Germany', 'Canada', 'Sov 'Australia', 'Greece', 'china', 'Great Britain', 'Brazil', 'Japan') </pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557) 9, 530, 462, 949) iet Union', 'South Korea', 'Spain', 'United States',</pre>
190 197 198 1902 2001 2002 2003 2004 2005 2007 2008 2009 2111 21212 2132 214 21212 2223 2224 2220 2211 21212 2223 2224 2220 2221 2223 2224 2220 2231 2223 2232 2333 2344 2356 237 238	<pre>Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.975) # Not including NL_France quantile(Y_star_w_unc,0.975) # Star_star = rpois(S,NI_France*lambda_star) hist(Y_star) Y_star_star) Y_star_star) Y_star_star) Y_star_star) Y_star_star) # Plot with and without NL_France uncertainty in PPD plot(NuLL, xlim=c(0,120), ylim=c(0,0.05), ylab="PPD", xlab="Medals won") dens_w_NL_unc = density(Y_star_w_unc) dens_w_NL_unc = density(Y_star) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) lines(dens_w_NL_uncSx,length(data)*dens_w_NL_uncSy,type="1",col='red',lwd=3) legend("topright",legend-c('with NL_France Unc', 'without NL_France Unc'), pch =) # Show results in a table result_table_PPD = matrix(c(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2.5.q Y_star_2.5.quantile,Y_star_97_5.quantile), nrow=2, ncol=4, byrow=1) result_table_PPD = c("Wean","std","quantile(2.5%)","quantile(97.5%) result_table_PPD - c("With NL_France Unce', 'without NL_France colnames(result_table_PPD) <- c("Wean","std","quantile(2.5%)","quantile(97.5%) result_table_PPD # MMM # Load in Data N0_per_country = c(129, 135, 94, 275, 208, 410, 175, 229, 941, 498, 140, 384, 30 NL_per_country = c(22, 36, 94, 011, 195, 33, 22, 275, 93, 16, 100, 51, 98, 80) Country_names = c('finland', 'traly', 'Mexico', 'West Germany', 'canada', 'Sow 'Australia', 'Greece', 'china', 'Great Britain', 'Brazil', 'Japan') </pre>	<pre>ummary statistics are computed rep(0,2),lwd-rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557) 9, 530, 462, 949) iet Union', 'South Korea', 'Spain', 'United States',</pre>
190 197 198 1902 2001 2002 2003 2004 2005 2007 2008 2009 2111 21212 2132 214 215 217 218 2201 2223 2224 2223 2224 2223 2230 2312 2332 234 2378 238 238 238 239	<pre>Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_25_guantile = quantile(Y_star_w_unc,0.975) # Not including NL_France uncertainty in the PPO, MC sampling is performed and s lambda_star = rgamma(S,Y0_France+a,N0_France+b) NL_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_star = rpois(S,N1_France*lambda_star) hist(Y_star) Y_star_sd = sd(Y_star) Plot(NuLL, xlim=c(0,120, ylim=c(0,0.05), ylab="PPO", xlab="Medals won") dens_w_N1_unc = density(Y_star_w_unc) dens_w_N1_unc = density(Y_star) lines(dens_w_N1_uncSx,length(data)*dens_w_N1_uncSy,type="1",col='red',lwd=3) lines(dens_w_N1_uncSx,length(data)*dens_w_N1_uncSy,type="1",col='red',lwd=3) lines(dens_w_N1_uncSx,length(data)*dens_w_N1_uncSy,type="1",col='red',lwd=3) legend("topright",legend-c('with N1_France Uncc', without N1_France Unc'), pch =) # Show results in a table result_table_PPD = matrix(C(Y_star_w_unc_mean,Y_star_w_unc_sd,Y_star_w_unc_2.5.q Y_star_2.5.quantile,Y_star_97_5.quantile), nrow=2, ncol=4, byrow=T) rownames(result_table_PPD) <- c("With N1_France Uncc', "without N1_France clinames(result_table_PPD) <- c("With N1_France Uncc', "uthout N1_France clinames(result_table_PPD) <- c("Watan","std","quantile (2.5%)","quantile (97.5%) result_table_PPD ##### # d d ##########################</pre>	<pre>ummary statistics are computed rep(0,2),lwd-rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557) 9, 530, 462, 949) iet Union', 'South Korea', 'Spain', 'United States',</pre>
190 197 198 1902 2001 2002 2003 2004 2005 2007 2008 2009 2111 21213 214 215 217 218 2201 2223 2224 2223 2230 2312 2332 2344 2356 2378 2378 2378 2378 2378 2378 2378 2378 2378 2378 2378 2378 2378 2378 2378 2374	<pre>Y_star_y_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_97_S_quantile = quantile(Y_star_w_unc,0.975) # Not including NI_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(S, Y0_France+a, N0_France+b) NI_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star = rpois(S,NL_France*lambda_star) hist(Y_star) Y_star_sda = sd(Y_star) Y_star_sda = sd(Y_star) Y_star_S</pre>	<pre>ummary statistics are computed rep(0,2),lwd-rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd uncertainty") 4, 236, 557) 9, 530, 462, 949) iet Union', 'South Korea', 'Spain', 'United States',</pre>
190 197 198 1902 2001 2002 2003 2005 2007 2009 2111 21212 2132 214 215 217 218 219 2211 21212 2212 2223 2224 2230 2312 2323 2334 2337 2344 2421 2412 2424	<pre>Y_star_y_unc_25_guantile = quantile(Y_star_w_unc,0.025) Y_star_w_unc_97_S_quantile = quantile(Y_star_w_unc,0.975) /* Not including NI_France uncertainty in the PPD, MC sampling is performed and s lambda_star = rgamma(5, Y0_France+a, N0_France+b) NI_France = fitScoefficients[1] + fitScoefficients[2]*N0_France Y_star_star_mean = mean(Y_star) Y_star_star_star_star_star_atar_atar_atar</pre>	<pre>ummary statistics are computed rep(0,2),lwd=rep(2,2),col=c('red','blue'), cex=1, pt.cex = 1 uantile,Y_star_w_unc_97_5_quantile,Y_star_mean,Y_star_sd Uncertainty") 4, 236, 557) 9, 530, 462, 949) iet Union', 'South Korea', 'Spain', 'United States', </pre>

245	# Define vectors	
246	r_mean_per_country = rep(0,length(Country_names))	
247	r_std_per_country = rep(0,length(Country_names))	
248	r_2_5_quantile_per_country = rep(0,length(Country_names))	
249	r_97_5_quantile_per_country = rep(0,length(Country_names))	
250		
251	r_MC_list = list()	
252		
252	# Plot nostariors of r for all countries and save summary statistics in vectors	
255	alat(will views)	
2 3 4	proc(NoLL, XIIm=c(0,3), yIIm=c(0,4), yIab= bensity, XIab= r varue)	
255	colors = c(`blue`,`red`,`green`,`orange`,`brown`,`purple`,`magenta`,`peachpuff`,'grey`,`black`,`darkseagreen`,`darksalmon`,`pink`,`khak`	
	,'lightblue')	
256 -	<pre>for (i in 1:length(Country_names)){</pre>	
257	lambda_0_MC_per_country <- rgamma(5,Y0_per_country[i]+a.N0_per_country[i]+b)	
258	lambda 1 MC per country <- rgamma(S X1 per country[i]+a N1 per country[i]+b)	
250	r we construct and a lambda i we not construct and a we not construct	
239	$r_mc_per_country = ramoua_r_mc_per_country/ramoua_o_mc_per_country$	
260		
261	r_MC_list[i] = list(r_MC_per_country)	
262		
263	r_mean_per_country[i] = mean(r_MC_per_country)	
264	r std per country[i] = sd(r MC per country)	
265	r 2 5 quantile per country[i] = quantile(r MC per country 0,025)	
266	r of s quantitie per country[1] - quantile(WC per country 0.075)	
200	dans dansity(a NC per south) = quartere(t_nc_per_country,0.3/3)	
207	dens = dens (cyc_mc_per_country)	
268	lines(dens\$x,length(data)*dens\$y,type="1",col=colors[1],lwd=3)	
269 🔺		
	legend("topright",legend=Country_names[1:15], pch = rep(0,15),lwd=rep(2,15),col=colors, cex=1, pt.cex = 1)	
271		
272	# Show results in a table	
273	result table $r = matrix(c(r mean per country r std per country r 2.5 quantile per country r 97.5 quantile per country) prow_length$	
	(country), not state the country, student state country, student state	
	(country_names), ncol=4, byrow=+)	
	rownames(result_table_r) <- Country_names	
275	colnames(result_table_r) <- c("Mean","std","quantile (2.5%)","quantile (97.5%)")	
276	result_table_r <- as.table(result_table_r)	
	result_table_r	
278		
270		
2/9		
280	# only output result of pairings once, e.g. Finland vs USA is the same conclusion as USA vs Finland	
	Country_name_1 = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2)	
282	Country_name_j = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2)	
282 283	Country_name_] = rep(0,(length(Country_names)*length(Country_names))length(Country_names))/2) r_diff_mean_per_country = rep(0,(length(Country_names)*length(Country_names))length(country_names))/2)	
282 283 284	<pre>Country_name_j = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_mean_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2)</pre>	
282 283 284 285	<pre>Country_name_j = rep(0,(length(country_names)*length(Country_names)-length(Country_names))/2) r_diff_mean_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2 5_guantile_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2)</pre>	
282 283 284 285 286	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2)</pre>	
282 283 284 285 286 286 287	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_97_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_97_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_or_sountry = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_or_sountry = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2)</pre>	
282 283 284 285 286 287 288	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_msq_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_st_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2)</pre>	
282 283 284 285 286 287 288 288	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2.5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_97_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per</pre>	
282 283 284 285 286 287 288 288 289	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, length(Country_names)*length(Country_names)+length(Country_names)+length(Country_names)+length(Country_names)+length(Country_names)+lengt</pre>	
282 283 284 285 286 287 288 289 290	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2.5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2.5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i_nn:length(Country_names)){</pre>	
282 283 284 285 286 287 288 289 290 290	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(country_names))/2) r_diff_s7_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_s7_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_s7_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ // for (j in 1:length(Countr</pre>	
282 283 284 285 286 287 288 289 290 290 291 292	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (j in 1:length(Country_names)){ if (i=j) { if (i=j) {</pre>	
282 283 284 285 286 287 288 289 290 291 292 293	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_s_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_s_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_s_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ if (i=j) { next } }</pre>	
282 283 284 285 286 287 288 289 290 290 291 292 293 294	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, length(Country_names)*length(Country_names)-length(Country_names)/2) r_diff_decision_per_country = rep(0, length(Country_names)*length(Country_names)+length(Country_names)/2) r_diff_decis</pre>	
282 283 284 285 286 287 288 289 290 291 292 293 294 295	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_sea_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_sf_2_s_quantile_per_country = rep(0, (length(Country_names)*length(country_names)-length(Country_names))/2) r_diff_sf_s_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_state_st</pre>	
282 283 284 285 286 287 288 289 290 291 292 293 294 295 295	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ if (i=j) { next } if (i>j) { next } } }</pre>	
282 283 284 285 286 287 288 290 291 292 293 294 295 295	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_sea_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_sf_2_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sf_s_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_stature_differences = 0 for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ nmext</pre>	
282 283 284 285 286 287 288 290 291 292 293 294 295 295 297	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_mean_per_country = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country_names)/2 r_diff_decision_per_country_names)/2 r_diff_decision_per_country_names)/2 r_diff_decision_per_country_names)/2 r_diff_decision_per_country_names/2 r_diff_decision_per_country_names/2 r_diff_decision_per_country_names/2 r_diff_decision_per_country_names/2 r_diff_decision_per_countr</pre>	
282 283 284 285 286 287 288 289 290 291 292 293 294 295 295 296 297 298	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_s7_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, length(Country_names)*length(Country_names))/2) r_diff_std_per_country_names)/2 r_diff_std_per_country_names/std_per_country_names/std_per_country_names)/</pre>	
282 283 284 285 286 287 288 289 291 292 293 294 295 296 297 298 297	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_gd_scapartile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country_names)){ index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ if (i=j) { next } if (i>j) { next } country_names[j] country_na</pre>	
282 283 284 285 286 287 288 290 291 293 294 295 295 296 297 298 299 300	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ next } if (i=j) { next } if (i=j) { country_names[j] country_names[j]</pre>	
282 283 284 285 286 287 288 290 291 292 293 294 295 296 297 298 299 297 298 299 300 301	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country_names)){ index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ if (i=j) { next } if (i>j) { next[} country_name_i[index] = Country_names[i] country_name_i[index] = country_names[i] country_name_i[index] = country_names[i] r_diff_mean_per_country[index] = mean(r_Mc_list[[i]]-r_MC_list[[j]]) r_diff_mean_per_country[index] = mean(r_Mc_list[[i]]) </pre>	
282 283 284 285 286 287 288 290 291 292 293 293 294 295 296 297 298 299 300 300 300 302	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ if (i=j) { next } if (i=j) {</pre>	
282 283 284 285 286 287 288 290 293 294 293 294 295 296 297 298 299 299 300 301 302 303	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_s7_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_s7_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names)/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names)/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)/2) r_diff_decision_per_country = rep(0, (length</pre>	
282 283 284 285 286 287 288 290 290 293 294 295 296 297 297 298 299 300 301 302 303 303	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ if (i=j) { next } if (i>j) { rext } if (i>j) { rext } country_name_i[index] = country_names[j] r_diff_std_per_country[index] = mean(r_Mc_list[[i]]-r_Mc_list[[j]]) r_diff_std_per_country[index] = mean(r_Mc_list[[i]]-r_Mc_list[[j]]) r_diff_std_per_country[index] = guantile(r_Mc_list[[i]]-r_Mc_list[[j]],0.025) r_diff_decision_per_country[index] = ifelse(guantile(r_Mc_list[[i]]-r_Mc_list[[i]],0.975) r_diff_decision_per_country[index] = ifelse(guantile(r_Mc_list[[i]]-r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]],0.925)*guantile(r_Mc_list[[i]]</pre>	75)
282 283 284 285 286 287 288 290 291 292 293 294 295 296 297 295 296 297 296 297 300 301 302 303 304	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_sf_2_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sf_s_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country_names)]{ for (j in 1:length(Country_names)){ if (i=j) { next } if (i=j) { rediff_mean_per_country[index] = country_names[i] country_names[i] country_names[i] country_names[i] country_names[i] r_diff_std_per_country[index] = sd(r_MC_list[[i]]-r_MC_list[[j]], 0.025) r_diff_st_squantile_per_country[index] = quantile(r_MC_list[[i]]-r_MC_list[[j]], 0.025) r_diff_decision_per_country[index] = ifelse(quantile(r_MC_list[[i]]-r_MC_list[[i]], 0.025)*quantile(r_MC_list[[i]]-r_MC_list[[i]], 0.025)*quantile(r_MC_list[[i]]-r_MC_list[[i]], 0.025)*quantile(r_MC_list[[i]]-r_MC_list[[i]], 0.025)*quantile(r_MC_list[[i]]-r_MC_list[[i]], 0.025)*quantile(r_MC_list[[i]]-r_MC_list[[i]], 0.025)*quantile(r_MC_list[[i]], 0.025)*quan</pre>	75)
282 283 284 285 286 287 288 290 291 292 293 294 295 294 295 297 298 299 299 300 301 302 303 304 305	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_sen_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_decision_per_country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ next if (i=j) { next } country_name_i[index] = Country_names[i] country_name_i[index] = country_names[i] country_name_i[index] = country_names[i] country_name_i[index] = country[index] = quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_97_5_quantile_per_country[index] = quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_97_5_quantile_per_country[index] = ifelse(quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_decision_per_country[index] = ifelse(quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_decision_per_country[index] = ifelse(quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_decision_per_country[index] = ifelse(quantile(r_MC_list[[i]]-r_MC_list[[i]],0.025)</pre>	75)
282 283 284 285 286 287 288 290 290 292 293 299 294 295 297 298 299 300 301 302 303 304 305 \$306	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_sf_2_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sf_3_s_quantile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ if (i=j) { nextl } for (j in 1:length(dex) = country_names[j] country_name_j[index] = country_names[j] country_names[j] country_names[j] country_names[j] country_names[j] country[index] = sd(r_MC_list[[j]]-r_MC_list[[j]]),0.025) r_diff_ss_squantile_per_country[index] = quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]]-r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],</pre>	75)
282 283 284 285 286 287 288 290 290 292 293 294 295 294 295 294 295 297 298 300 301 302 303 304 305 306	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_sea_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_97_5_quartile_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country_names)){ for (i in 1:length(Country_names)){ if (i=j) { next[</pre>	75)
282 283 284 285 286 287 290 290 292 293 294 295 297 298 297 297 300 301 302 301 302 303 304 305 306	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country_names)[for (j in 1:length(Country_names)]{ rot (j in 1:length(country_names)]{ rountry_names[j] rot (j in 1:length(country</pre>	75)
282 283 284 285 286 287 288 290 290 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))/ength(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_s7_5_quantile_per_country = rep(0, (length(Country_names)*length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names))/2) if (i=j) { next</pre>	75)
282 283 284 285 286 287 291 293 294 295 297 293 294 295 297 298 299 300 301 302 303 304 305 306 307 4 308 309	<pre>Country_name_j = rep(0,(length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0,(length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_25.cguantile_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)]{ r_diff_2.5_quantile_gen_country[index] = ad(r_MC_list[[j]],r_MC_list[[j]])r_MC_list[[j]],0.025) r_diff_2.5_quantile_gen_country[index] = quantile(r_MC_list[[i]]-r_MC_list[[i]],r_MC_list[[j]],0.025)*quantile(r_MC</pre>	75)
282 283 284 285 286 287 288 289 290 291 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 300 308 309 310	<pre>Country_name_j = rep(0,(length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_2.5_quantile_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_(length(Country_names)){ for (i in l:length(Country_names)){ for (j in l:length(Country_names)){ if (i=j) { next } if (i=j) { next } country_name_j[index] = country[index] = mean(r_MC_list[[i]])-r_MC_list[[j]]) r_diff_std_per_country[index] = quantile(r_MC_list[[j]]) r_diff_std_per_country[index] = quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_decision_per_country[index] = quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_decision_per_country[index] = quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025) r_diff_decision_per_country[index] = ifelse(quantile(r_MC_list[[i]]-r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.025)*quantile(r_MC_li</pre>	75)
282 283 285 286 287 288 289 290 291 292 293 294 295 295 295 297 298 200 301 302 303 304 305 303 304 305 306 307 308 309 309 301 201 201 201 201 201 201 201 201 201 2	<pre>Country_name_j = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)]{ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)]{ for (j in</pre>	75)
282 283 284 285 286 287 287 292 291 292 293 294 295 296 297 298 299 300 302 303 304 305 306 307 308 309 4 305 307 308 309 300 309 4 305 309 300 309 300 300 300 300 300 300 300	<pre>Country_name_j = rep(0,(length(Country_names)*length(Country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(Country_names))-length(Country_names))/2) r_diff_2.5_quantile_per_country = rep(0,(length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_2.5_quantile_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)]{ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)]{ rot (j names) = (j nam</pre>	75)
282 283 284 285 286 287 288 290 291 292 293 294 295 297 298 297 297 298 209 301 302 303 304 305 306 307 308 307 307 308 307 307 307 307 307 307 307 307 307 307	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_decision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)]{ for (j in 1:length(Country_names]]{ for (j in 1:l</pre>	75)
282 283 284 285 286 287 288 290 291 292 293 294 295 296 297 297 297 297 297 300 301 302 303 304 305 306 307 308 309 301 305 306 307 308 309 301 305 306 307 308 309 301 305 306 307 308 309 301 305 306 307 308 309 301 305 306 307 308 309 307 308 309 307 308 309 309 300 300 300 300 300 300 300 300	<pre>Country_name_j = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0, (length(country_names)*length(Country_names))-length(Country_names))/2) r_diff_ds_squantile_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_dscision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_dscision_per_country = rep(0, (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_dscision_per_country = rep(0, (length(Country_names)*length(Country_names))-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (j in 1:length(Country_names)){ for (j in 1:length(Country_names)){ if (i=j) { next } if (i=j) { next } country_name_j[index] = country_names[j] country_name_j[index] = country[index] = quantile(r_Mc_list[[j]]) r_diff_dscision_per_country[index] = quantile(r_Mc_list[[j]]) r_diff_dscision_per_country[index] = quantile(r_Mc_list[[j]])-r_Mc_list[[j]],0.025) r_diff_dscision_per_country[index] = quantile(r_Mc_list[[j]]-r_Mc_list[[j]],0.025) r_diff_dscision_per_country[index] = fielse(quantile(r_Mc_list[[j]],-r_Mc_list[[j]],0.025) r_diff_dscision_per_country[index] = differences + 1 } for (ringth(Country_name_i,country_name_i,country_name), r_diff_est_per_country, r_diff_2_5_quantile_per_country[index] = fielse(quantile(r_Mc_list[[j]],-r_Mc_list[[j]],0.025) * 0, { number_of_true_differences = number_of_true_differences + 1 } for (results in a table. If final column has True then there is a difference in r for the two countries results in a table. If final column has True then there is a difference in r for the two countries result_table_r_diff = matrix(CCountry_name_i, country, name_i, r_diff_man_per_country, r_diff_2_5_quantile_per_country r_diff_decision_per_c</pre>	75)
282 283 284 285 286 287 288 290 292 292 292 292 293 294 295 297 298 299 300 301 302 303 304 305 306 307 308 308 308 308 309 310 311 311 311 314	<pre>Country_name_j = rep(0,(length(country_names)*length(country_names))/2) r_diff_ext_per_country = rep(0,(length(country_names)*length(country_names))/2) r_diff_cst_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_cst_cs_quantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_cst_cs_quantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_cst_cs_quantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_cst_csion_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_cst_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) for (j in 1:length(country_names)){ for (j in 1:length(country_names)){ for (j in 1:length(country_names)]{ for (j</pre>	75) try
282 283 284 285 286 287 288 290 291 293 294 295 297 298 297 298 297 300 301 302 303 304 305 306 307 308 309 301 310 311 312 313 314 315	<pre>Country_name_j = rep(0,(length(country_names)*length(country_names))/2) r_diff_exa_per_country = rep(0,(length(country_names)*length(country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(country_names)){ for (j in 1:length(country_names)){ if (i=j) { next</pre>	75) try
282 283 284 285 286 287 288 290 290 291 293 294 293 294 295 295 297 295 297 295 297 300 301 302 303 304 306 307 306 307 310 311 311 311 314 315	<pre>Country_name_j = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_std_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ if (1=j) { next } for (j in 1:length(Country_names)){ if (1=j) { next } for (j in 1:length(Country_names)]{ country_index] = country[index] = duantile(r_MC_list[[j]]) r_diff_std_per_country[index] = mean(r_MC_list[[j]]-r_MC_list[[j]]) r_diff_std_per_country[index] = quantile(r_MC_list[[j]]-r_MC_list[[j]],0.025) r_diff_dectsion_per_country[index] = quantile(r_MC_list[[j]]-r_MC_list[[j]],0.025) r_diff_std_per_country[index] = quantile(r_MC_list[[j]]-r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.975) r_diff_dectsion_per_country[index] = quantile(r_MC_list[[j]]-r_MC_list[[j]],0.025)*quantile(r_MC_list[[j]],0.975) > 0) { number_of_true_differences = number_of_true_differences + 1 } } } } # Show results in a table. If final column has True then there is a difference in r for the two countries result_table_r_diff < < as. table(result_table_r_diff) </pre>	75) try
282 283 284 285 286 287 288 299 291 292 293 299 297 297 297 297 297 297 297 297 299 300 302 301 302 303 304 305 307 308 309 300 301 305 307 308 309 301 305 307 308 309 301 309 311 311 311 311 311 317	<pre>Country_name_j = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_act_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sct_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sct_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sct_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sct_per_country = rep(0,(length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names)){ for (j in 1:length(Country_names)){ if (i=j) { rest if (i=j) { rest if (i=j) { rest } country_name_j[index] = country_index] = garant[er_Mc_list[[j]]) r_diff_decision_per_country[index] = garant[er_Mc_list[[j]]) r_diff_mean_per_country[index] = garant[er_Mc_list[[j]]) r_diff_decision_per_country[index] = garant[er_Mc_list[[i]]-r_Mc_list[[j]],0.023) r_diff_decision_per_country[index] = garantile(r_Mc_list[[i]]-r_Mc_list[[j]],0.023) r_diff_decision_per_country[index] = ifelse(quantile(r_Mc_list[[i]]-r_Mc_list[[j]],0.023) r_diff_decision_per_country[index] = ifelse(quantile(r_Mc_list[[i]]-r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[[j]],0.023)*quantile(r_Mc_list[</pre>	75) try
282 283 284 285 286 287 288 299 290 293 294 293 294 295 295 296 297 297 295 297 296 297 297 295 297 296 297 297 296 297 297 297 297 297 297 297 297 297 297	<pre>Country_name_j = rep(0,(length(country_names)*length(Country_names)-length(country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(Country_names)-length(country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(Country_names)-length(country_names))/2) r_diff_st_std_per_country = rep(0,(length(country_names)*length(Country_names)-length(country_names))/2) r_diff_st_st_guantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_st_st_guantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_st_st_guantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_st_st_guantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_st_guantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_st_guantile_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_st_guantile_per_country_names) { ind(i i i :length(country_names)) { ind(i i i :length(country_names)) { ind(i i i :length(country_names)) { if (i=j) { next] } country_name_j[index] = country_names[j] country_name_j[index] = country[index] = aquartile(r_MC_list[[j]], n.MC_list[[j]], 0.025) r_diff_st_der_country[index] = sife sequantile(r_MC_list[[j]], n.MC_list[[j]], 0.025) r_diff_st_st_guantile_per_country[index] = ifelse(quantile(r_MC_list[[i]], r_MC_list[[j]], 0.025) r_diff_st_st_guantile_per_country[index] = ifelse(quantile(r_MC_list[[i]], r_MC_list[[j]], 0.025) r_diff_st_st_guantile_per_country[index] = ifelse(quantile(r_MC_list[[i]], r_MC_list[[i]], 0.025) r_diff_st_st_guantile_per_country[index] = ifelse(quantile(r_MC_list[[i]], r_MC_list[[i]], 0.025) r_diff_st_st_guantile_per_country[index] = ifelse(quantile(r_MC_list[[i]], r_M</pre>	75) try
282 283 284 285 286 287 288 299 290 292 293 294 295 296 297 298 299 300 301 303 304 305 306 307 308 307 308 307 308 309 310 311 312 313 314 315 317 318	<pre>Country_name_j = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_std_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_std_ston_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) r_diff_std_ston_per_country = rep(0,(length(country_names)*length(country_names)-length(country_names))/2) index = 1 number.of_true_differences = 0 for (i in 1:length(country_names))[for (i in 1:length(country_names))[</pre>	75) try
282 283 285 286 287 288 290 293 294 293 294 295 297 298 299 201 297 298 201 301 302 303 304 305 306 308 309 301 311 312 313 314 315 318 319 290	<pre>Country_name_j = rep(0;(length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0;(length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_sy_5_quantile_per_country = rep(0; (length(Country_names)*length(Country_names))-length(Country_names))/2) r_diff_sy_5_quantile_per_country = rep(0; (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sy_5_quantile_per_country = rep(0; (length(Country_names)*length(Country_names)-length(Country_names))/2) r_diff_sy_5_quantile_per_country = rep(0; (length(Country_names)*length(Country_names)-length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1:length(Country_names))(if (i=j) { next[</pre>	75) try
2822 2833 2844 2855 2866 2877 2828 2901 2929 2929 2929 2929 2929 2929 2929	<pre>Country_name_j = rep(0;(length(Country_names)' length(Country_names))-length(Country_names))/2) r_diff_std_per_country = rep(0;(length(Country_names)' length(Country_names))/2) r_diff_std_per_country = rep(0;(length(Country_names)' length(Country_names))/2) r_diff_sy_s_quantile_per_country = rep(0; (length(Country_names)' length(Country_names))/2) r_diff_sy_s_quantile_per_country = rep(0; (length(Country_names)' length(Country_names))/2) r_diff_sy_s_quantile_per_country = rep(0; (length(Country_names)) length(Country_names))/2) index = 1 number_of_true_differences = 0 for (i in 1: length(Country_names))[if (i = 1) length(Country_names)][if (i = 1) length(Country_names)][if (i = 1) length(Country_names)][country_name_j[index] = country[index] = man(r_M_list[[i]]) r_diff_std_per_country[index] = man(r_M_list[[i]])-r_M_list[[j]]) r_diff_std_per_country[index] = man(r_M_list[[i]])-r_M_list[[j]]) r_diff_std_per_country[index] = ide(quantile(r_M_list[[i]])-r_M_list[[j]]),0.023) r_diff_decision_per_country[index] = ide(quantile(r_M_list[[i]])-r_M_list[[j]],0.023) r_diff(quantile(r_M_list[[i]])-r_M_list[[i]])-r_M_list[[j]],0.023) r_diff_decision_per_country[index] = ide(quantile(r_M_list[[i]])-r_M_list[[j]],0.023) r_diff_decision_per_country[index] = ide(quantile(r_M_list[[i]])-r_M_list[[j]],0.023) r_diff_decision_per_country[index] = ide(quantile(r_M_list[[i]])-r_M_list[[j]],0.023) r_diff_decision_per_country[index] = ide(quantile(r_M_list[[i]])-r_M_list[[j]],0.023) r_diff_decision_per_country[index] = ide(quantile(r_M_list[[i]])-r_M_list[[j]]),0.023) r_diff_decision_per_</pre>	75) try