TREBALL DE FI DE GRAU

Grau en Estadística

Títol: Desenvolupament d'una aplicació per a l'aprenentatge i la docència de l'Estadística Bayesiana

Autor: Núria Busquets

Director: Xavier Puig i Lluís Marco

Departament: Estadística i Investigació Operativa (UPC)

Convocatòria: Febrer 2016

Resum

Des de començaments del 2015, el departament d'Estadística i Investigació Operativa de la Universitat Politènica de Catalunya du a terme un projecte per desenvolupar una aplicació de lliure distribució per facilitar l'aprenentatge de l'estadística en l'educació secundària. L'aplicació s'anomena *StatClip*. Basant-se en la interfície d'usuari creada, en aquest projecte es desenvolupa *BayesClip,* una nova aplicació lliure que faciliti l'aprenentatge de l'estadística Bayesiana en cursos d'iniciació d'aquesta branca. Pel desenvolupament de *BayesClip* s'estudiarà el llenguatge de programació *Shiny*, conceptes d'estadística Bayesiana, s'elaborarà una recerca d'aplicacions ja existents, es decidiran i implementaran els elements de l'aplicació i finalment es testarà en alumnes del grau d'estadística per determinar el seu impacte. L'aplicació es pot trobar a https://noctilabium.shinyapps.io/BayesClip/.

Paraules clau: Aplicació *web,* Docència, Estadística Bayesiana, Programació reactiva, *R*, *Shiny.*

Abstract

At the beginning of 2015, the Statistics and Operation Research department of Universitat Politècnica de Catalunya started a project to develop a free app to help in the task of learning statistics at high school. The app is called *StatClip*. Based on the user interface of this app, a brand new free app is developed in this project, called *BayesClip*, in order to help in the task of learning Bayesian statistics in introductory courses of this branch of statistics. In the development of this app, the *Shiny* programming language will be studied as well as Bayesian statistics concepts, a research of existing applications will be carried out, the design and implementation of the app will be analysed and finally, the app will be tested in university students of degree in statistics in order to determine its impact among learners. The app can be found at https://noctilabium.shinyapps.io/BayesClip/.

Keywords: Bayesian Statistics, *R,* Reactive programing, *Shiny,* Teaching, web application.

Classificació AMS (American Mathematical Society)

62-04: *Explicit machine computation and programs (not the theory of computation or programming)*

62-09: *Graphical methods*

62F15 *Bayesian inference*

Índex

1. Introducció

Des de començaments del 2015, el departament d'Estadística i Investigació Operativa de la Universitat Politènica de Catalunya duu a terme un projecte per desenvolupar una aplicació de lliure distribució per facilitar l'aprenentatge de l'estadística en l'educació secundària. L'aplicació s'anomena *StatClip i* actualment es pot trobar a https://noctilabium.shinyapps.io/StatClip/. Basant-se en la interfície d'usuari creada, en aquest projecte es proposa desenvolupar una nova aplicació lliure que faciliti l'aprenentatge de l'estadística Bayesiana en cursos d'iniciació en aquesta disciplina.

L'objectiu és crear una aplicació que permeti a l'usuari concentrar-se en aprendre els conceptes de l'estadística Bayesiana i no en aprendre com elaborar codi. Per aquesta raó l'aplicació es desenvoluparà amb codi *Shiny,* ja que quan es va iniciar l'aplicació anterior (*StatClip)* es va fer una recerca d'opcions i *Shiny* va resultar ser la millor. El *Shiny* és una llibreria del programa estadístic *R* que permet implementar aplicacions *web* interactives i fàcils d'utilitzar. L'usuari podrà utilitzar l'aplicació sense la necessitat de saber *R.*

Els objectius d'aquest projecte són:

- Familiaritzar-se amb l'aplicació actualment en desenvolupament i dissenyar la integració del nou mòdul.
- Identificar les opcions que ha de tenir l'aplicació, en base als requeriments dels usuaris.
- Dissenyar la interfície.
- Implementar l'aplicació.
- Testar l'aplicació amb estudiants.

Així doncs, per assolir els objectius proposats, el projecte es divideix en una secció on s'explica què és el *Shiny*, així com la descripció dels seus elements i com s'elabora una aplicació, una secció d'estadística Bayesiana on s'expliquen els seus elements i algoritmes d'estimació, una secció anomenada *StatClip* on es descriuen els elements de l'aplicació ja creada i l'estructuració d'aquesta, una secció de recerca d'altres aplicacions web Bayesianes ja creades, seguidament la secció més important on s'explica el procés de creació de la nova aplicació i finalment una secció on s'analitza l'impacte que ha tingut aquesta.

StatClip va ser dissenyada de forma que pogués ésser ampliada fàcilment. Aleshores, el nou mòdul implementat en aquest projecte s'anomenarà *BayesClip.*

S'ha dedicat molt d'esforç i treball en aquest projecte. Per la construcció de *BayesClip* han sigut necessàries moltes hores de programació i de treball de càlcul i el resultat obtingut ha sigut molt agradable i satisfactori. L'aplicació final es pot observar a: https://noctilabium.shinyapps.io/BayesClip/.

2. Introducció al Shiny

Shiny és una extensió de *R* que va ser creada al 2012 per *R-Studio* i que serveix per crear aplicacions interactives orientades al anàlisis i visualització de dades a través de *R*. **[1]**

Un dels seus millors avantatges és que per construir les aplicacions no és necessari tenir coneixements de codi web, com ara *HTML* o *JavaScript*, tot i que si es desitja es poden introduir directament codis en aquest llenguatge mitjançant les indicacions necessàries. El que fa *Shiny* per si sol és traduir les funcions d'*R* a codi *HTML* i d'aquesta forma construir l'aplicació.

Shiny serveix per transmetre coneixement, funcions i algoritmes a usuaris no familiaritzats amb codi *R* ja que no es necessari escriure. Solament fent un clic es canvien els arguments i els resultats s'actualitzen immediatament.

Les aplicacions elaborades amb *Shiny* es poden executar en un terminal de forma local o mitjançant un navegador si l'aplicació està penjada en un servidor. Per executar les aplicacions de forma local, és necessari utilitzar el *R* i instal·lar-hi el paquet *Shiny*

Pel que fa als suports d'ajuda, *Shiny* disposa d'una amplia gamma d'articles, referències per les funcions, una galeria amb exemples d'aplicacions i tutorials en format vídeo o de forma escrita proporcionats pels desenvolupadors. Mitjançant aquests recursos l'usuari pot aprendre a crear aplicacions **[2]**.

Seguidament es veurà com funciona internament *Shiny*, es descriuran els elements per crear una aplicació *Shiny*, es mostrarà com posicionar-los visualment en l'aplicació i finalment es detallarà el codi per construir una aplicació bàsica amb *Shiny.*

2.1 Com funciona el Shiny

Una aplicació *Shiny* consta de dues components: una interfície d'usuari definida en un arxiu anomenat *UI.r*, i un script del servidor definit en un arxiu *server.r*. D'aquesta forma es classifiquen els elements de l'aplicació en: els elements responsables d'efectuar càlculs en l'arxiu *server.r* (taules, figures, gràfics, etc.), i els elements que l'usuari veurà (botons, llistes, barres lliscants, etc.), que defineixen l'aparença de l'aplicació mitjançant instruccions que es tradueixen a codi HTML en l'arxiu *UI.r*.

El *Shiny* utilitza objectes *input* i *output.* Els elements *input* són botons, barres lliscants, etc. Es defineixen mitjançant l'argument *inputId* de les funcions *checkboxInput*, *selectInput*, entre altres. Per accedir al valor que pren un *input* s'utilitza *input$inputId.* Els elements output es defineixen mitjançant la comanda *output$nomelement*.

L'intercanvi d'informació de *input* i *output* entre les dues parts de l'aplicació s'elabora de forma automàtica, és a dir, quan l'usuari modifica el valor d'algun *input*, l'aplicació detecta el canvi i torna a executar els fitxers *Ui.R* i *server.R.* Shiny fa servir, doncs, l'anomenada *programació reactiva*, ja que permet construir una interfície que respon immediatament a qualsevol canvi que l'usuari elabori. De la mateixa forma, les seves funcions són funcions que permeten crear variables reactives, que són aquelles variables que es modifiquen cada vegada que es realitza un canvi en el *input.*

El *Shiny* també ofereix la possibilitat de que el codi no s'executi automàticament cada vegada que es modifica qualsevol *input*, ja que pot suposar un cost computacional excessiu. Mitjançant un botó*, update*, l'usuari pot manipular tots els *inputs* i no observar els resultats fins que es pressioni aquest botó.

2.2 Elements per crear una aplicació Shiny

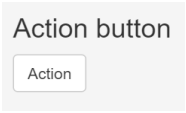
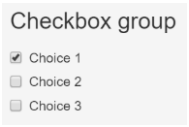
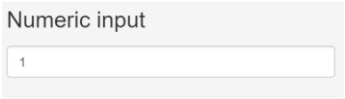
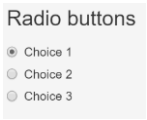
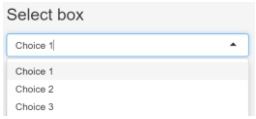
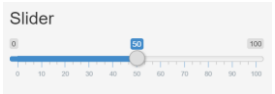
Una aplicació *Shiny* consta d'elements *input* i elements *output.* Els element *input* són necessaris per tal de que l'usuari introdueixi certes característiques que es visualitzaran en els *outputs*. Per exemple, el gràfic de la funció de densitat d'una $Normal$ (*output*) es modificarà mitjançant una barra lliscant (*input)* que mourà l'usuari.

Existeixen moltes opcions, *widgets,* per que l'usuari pugui veure i introduir *inputs*. La pàgina web de *Shiny* proporciona una galeria amb exemples i el codi específic necessari per introduir els *widgets* a l'aplicació **[3]**. En la Taula 2.1 s'observa un resum dels *inputs* que s'utilitzaran per construir l'aplicació

Mitjançant els valors dels input es creen *outputs*. Per crear i visualitzar *outputs* primerament s'ha de calcular l'*output* en el fitxer *server.r*, dins d'una funció *render*, i després indicar al fitxer *UI.r* l'aparença i posició del resultat. A la Taula 2.2 s'indiquen les correspondències entre les funcions per elaborar un *output* que s'utilitzaran en la creació de l'aplicació.

Existeix una funció anomenada *reactive*, que s'utilitza en el fitxer *server.r* i serveix per introduir càlculs que solament es recalcularan quan es modifiquin els valors dels *inputs* utilitzats en aquesta funció. D'aquesta forma, quan es modifica un *input* aliè al càlcul de dins de la funció *reactive*, aquesta no es tronarà a executar. D'aquesta forma es pot estalviar temps de computació si el càlcul és excessivament costós.

Taula 2.1 Aparença, codi i descripció dels *Widgets* utilitzats en *BayesClip*

Aparença	Codi
	`actionButton("action", label = "Action")`
Desencadena una acció quan es prem. Ja sigui mostrar un *output* o actualitzar-lo.	
	`checkboxGroupInput("checkGroup", label = h3("Checkbox group"),choices = list("Choice 1" = 1, "Choice 2" = 2, "Choice 3" = 3), selected = 1)`
Permet seleccionar varies opcions simultàniament.	
	`fileInput("file", label = h3("File input"))`
Permet seleccionar un arxiu del terminal i carregar-lo a l'aplicació.	
	`numericInput(inputId, label, value, min = NA, max = NA, step = NA)`
Permet introduir un valor numèric.	
	`radioButtons("radio", label = h3("Radio buttons"), choices = list("Choice 1" = 1, "Choice 2" = 2, "Choice 3" = 3), selected = 1)`
Permet seleccionar només una opció.	
	`selectInput("select", label = h3("Select box"), choices = list("Choice 1" = 1, "Choice 2" = 2, "Choice 3" = 3), selected = 1)`
Obre un desplegable amb diverses opcions. Només se'n pot seleccionar una.	
	`sliderInput("slider1", label = h3("Slider"), min = 0, max = 100, value = 50)`
S'utilitza per seleccionar un valor d'entre el rang de valors de la barra lliscant.	

Taula 2.2 Correspondències de les funcions internes de *BayesClip* entre els fitxers *server.r* i *UI.r*

Funció	***server.r***	***UI.r***
Per la representació de gràfics. El gràfic es crea i modifica, però no es guarda.	`renderPlot`	`plotOutput`
Per representar una matriu o un *data frame.*	`renderTable`	`tableOutput`
Tracta els *inputs*. En el cas particular de *BayesClip* permet amagar o mostrar elements quan sigui necessàri.	`renderUI`	`uiOutput`

2.2.1 Disposició dels elements

Un cop determinats els elements input i output és necessari especificar el lloc on es trobaran visualment a l'aplicació. Shiny disposa de diversos elements per organitzar els objectes, però ja que l'aplicació que es crearà serà sobre una plantilla *shinydashboard* (per la seva organització, aparença visual i varietat de colors) solament s'explicarà la disposició dels elements en aquesta. **[4]**

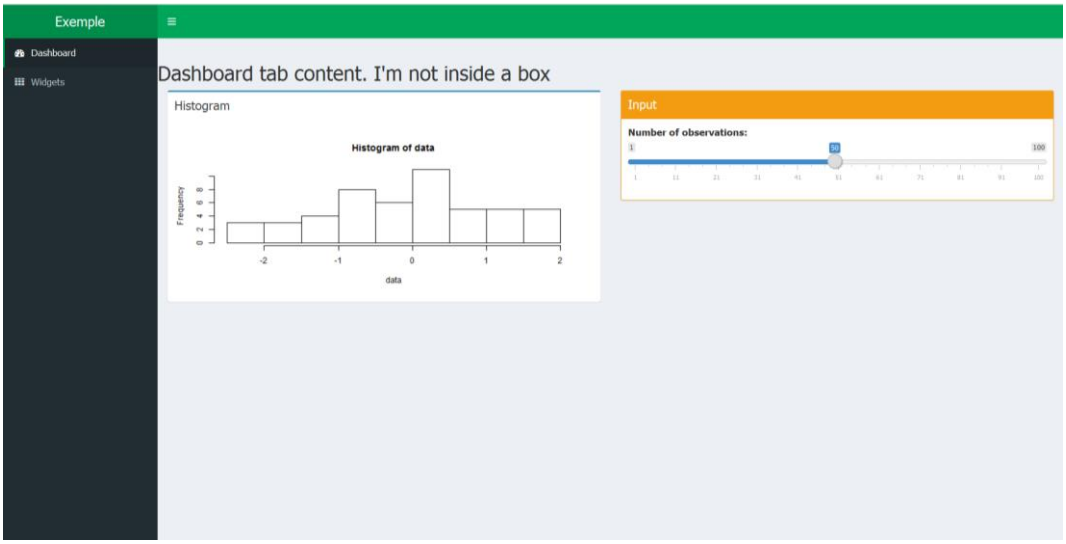


Figura 2.1 Exemple d'aplicació bàsica mitjançant la plantilla de *shinydashboard*. Secció *Dashboard*.

En la Figura 2.1 s'observa la plantilla de *shinydashboard,* que consta d'una capçalera, una barra lateral i un cos. En la capçalera, en aquest cas de color verd, s'hi troba el nom de l'aplicació, en la barra lateral els diferents apartats i subapartats de l'aplicació amb els seus corresponents cossos i en el cos s'hi troben els *inputs* i *output*. En l'espai del cos, es

posicionen tots els elements de l'aplicació i, opcionalment, es poden agrupar en seccions o *caixes*. Aquestes caixes poden ser simples o amb pestanyes, i amb diferents aparences.

En la Figura 2.1 s'aprecia una frase que no es troba dins de cap caixa, un histograma dins d'una caixa simple amb una franja de color blau i una barra lliscant dins d'una caixa amb capçalera de color taronja. Les caixes estan distribuïdes en fila al llarg de tota l'ampla del cos. Gràcies a la funció *fluidRow*, no es necessari determinar la mida concreta de cada caixa, sinó que és l'aplicació qui adapta la mida de cada caixa en funció de la grandària de la pantalla o finestra.

Una caixa amb pestanyes és molt útil per si no es té suficient lloc per tots els element d'un cos. Les caixes amb pestanyes s'anomenen *tabBox*, i cada pestanya, *tabPanel*. A la Figura 2.2 s'aprecia una caixa simple i una *tabBox,* que consta de dos *tabPanel*. En aquest cas els elements del cos estan posicionats en una columna de mida 4. La mida total del cos és de 12 unitats, essent possible només divisions enteres d'unitats. Per posicionar els element en columna s'utilitza la funció `column(width=4,...),` enlloc de *fluidRow*, on *width* determina les unitats d'amplada de la columna.

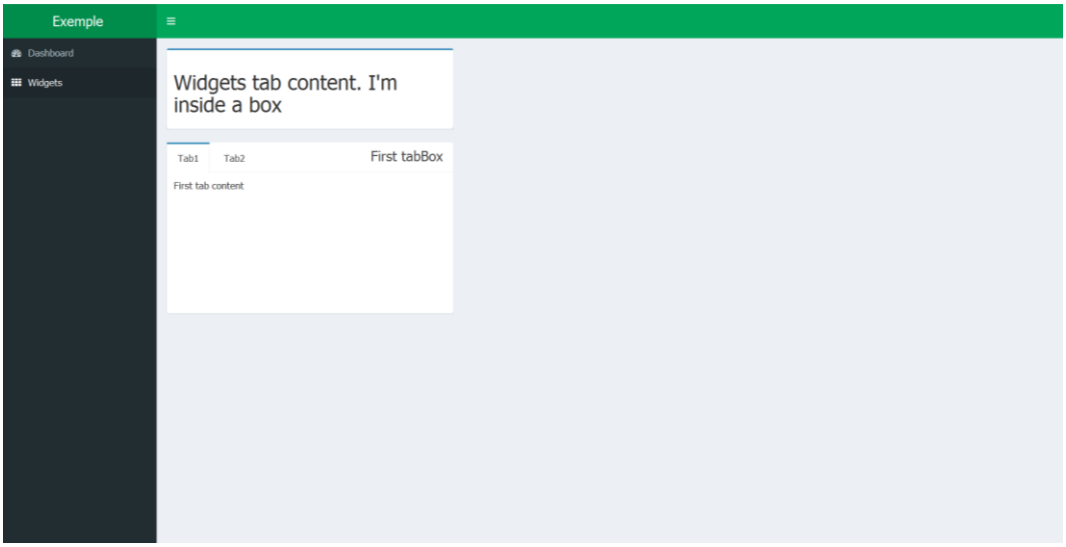


Figura 2.2 Exemple d'aplicació bàsica mitjançant la plantilla de *shinydashboard*. Secció *Widgets*.

Cada secció de la barra lateral és un *menuItem*. Aquestes seccions permeten classificar els cossos, és a dir, cada secció conté un cos diferent. Per enllaçar un cos amb una secció inicialment s'ha d'implementar el cos. Per implementar un cos s'introdueixen els seus elements dins d'una funció *tabItem* i s'identifica el cos mitjançant la comanda *tabName*. Finalment en cada secció de la barra lateral (*menuItem*) s'utilitza també la comanda *tabName* per indicar l'identificador del cos que es vol que contingui.

2.3 Com construir una aplicació Shiny

Cada aplicació *Shiny* necessita, almenys, un fitxer *UI.R* i un fitxer *server.r*. Opcionalment es poden complementar aquest fitxers amb d'altres que continguin funcions necessàries per l'aplicació, sempre i quan s'elabori la crida corresponent. Tots els fitxers necessaris per l'aplicació han d'estar localitzats en un mateix directori amb el nom de l'aplicació.

En les seccions anteriors s'han descrits els element que formen l'aplicació (els *inputs*, *outputs,* la plantilla de *shinydashboard* i les possibles formes de posicionar els elements al cos). A continuació es procedeix a descriure l'elaboració del codi.

Per tal de començar a construir una aplicació *Shiny* mitjançant una plantilla *shinydashboard* cal tenir els paquets necessaris instal·lats. Per instal·lar el paquet *Shiny* s'utilitza la comanda `install.packages("shiny")` i per utilitzar la plantilla *shinydashboard* s'utilitza la comanda `install.packages("shinydashboard")`.

Seguidament en l'arxiu *UI.r* s'escriu l'estructura simple de la plantilla, que és:

```
dashboardPage(
      dashboardHeader(),
      dashboardSidebar(),
      dashboardBody()
)
```

S'observa que tots els elements del fitxer *UI.r* s'han de separar per una coma. És a dir, després de cada element del fitxer *UI.r* es col·locarà una coma, excepte en l'últim element del conjunt.

En el *dashboardHeader* s'hi indicarà el nom de l'aplicació i el color de la capçalera. En el *dashboardSidebar* s'hi indicaran els diferents apartats de l'aplicació i la relació amb el corresponent cos i en el *dashboardBody* s'hi implementaran els diferents cossos.

A partir d'ara, per explicar l'elaboració d'un codi *Shiny* es seguirà la construcció de l'aplicació que s'observa en la Figura 2.1 i la Figura 2.2.

Per tal d'escollir el color de la capçalera de l'aplicació s'utilitza el paràmetre *skin*. Així doncs dins del *dashboardPage* s'hi implementa `skin = "green"` per tal de que la capçalera sigui de color verd, com s'observa en la Figura 2.1.

Seguidament, per indicar el títol de l'aplicació (que apareix a la part esquerra de la capçalera) s'utilitza el paràmetre *title* de la següent forma: `dashboardHeader(title = "Exemple")`.

Per especificar la barra lateral s'ha d'utilitzar la funció *sidebarMenu* dins de *dashboardSidebar*. A la funció *sidebarMenu* se li assigna un identificador i seguidament, per determinar els diferents apartats, s'utilitza *menuItem*. És a dir, a cada *menuItem* s'hi indica el nom i la icona de l'apartat i l'enllaç del cos corresponent. Per enllaçar el cos, s'utilitza l'identificador amb la

comanda *tabName*, és a dir, el cos té un identificador amb la comanda *tabName* i aleshores *menuItem* la crida mitjançant també *tabName*. A continuació es mostra l'especificació de la barra lateral de la Figura 2.1.Figura 2.1 Exemple d'aplicació bàsica mitjançant la plantilla de *shinydashboard*. Secció *Dashboard*.

```
dashboardSidebar(
    sidebarMenu(
      id = "tabs",
      menuItem("Dashboard", tabName = "dashboard",
               icon = icon("dashboard")),
      menuItem("Widgets", tabName = "widgets", icon = icon("th"))
    )
  ),
```

Seguidament s'ha d'implementar el cos de cada pestanya dins de *dashboardBody*. Els elements d'un cos estan implementats dins d'un *tabItem*. Finalment la funció *dashboardBody* conté una funció anomenada *tabItems* que conté la col·lecció de tots els *tabItem.* Per tant, l'estructura és:

```
dashboardBody(
    tabItems(
             tabItem(...),
             tabItem(...)
    )
),
```

La implementació del cos per l'apartat de *Dashboard* de la Figura 2.1 és:

```
tabItem(tabName = "dashboard",
          fluidRow(
            h2("Dashboard tab content. I'm not inside a box"),
            # Boxes need to be put in a row (or column)
            box(
             title = "Histogram", status = "primary",
             plotOutput("plot1")
            ),

            box(
             title = "Input", status = "warning", solidHeader = TRUE,
             sliderInput("slider", "Number of observations:", 1, 100, 50)
            )
          )
),
```

En la implementació anterior s'observa que la frase "*Dashboard tab content. I'm not inside a box*" no està situada dins de cap caixa (*box*) i que està dins d'una funció *h2*. Aquesta funció serveix per indicar la grandària del títol en una escala d'1 al 5, éssent 1 el títol més gran i 5 el títol més petit. També s'aprecia una caixa amb una franja de color blau (indicat amb l'opció *status*) que conté el gràfic de l'histograma anomenat *plot1* i la *caixa* de color taronja que conté la barra lliscant.

Seguidament la implementació del cos per l'apartat de *Widgets* de la Figura 2.2 és:

```
tabItem(tabName = "widgets",
          fluidRow(
            column(width=4,
              box(width=NULL,
                status = "primary",
                h2("Widgets tab content. I'm inside a box")
              ),
              tabBox(width=NULL,
                title = "First tabBox",
                id = "tabset1", height = "250px",
                tabPanel("Tab1", "First tab content"),
                tabPanel("Tab2", "Tab content 2")
              )
            )
          )
 )
```

S'aprecia que els elements estan posicionats en una columna de mida 4. En aquest cas la frase "*Widgets tab content. I'm inside a box"* es troba a l'interior d'una caixa. S'observa que la implementació de la caixa amb pestanyes es duu a terme mitjançant *tabBox*, i les pestanyes, les quals només contenen text, mitjançant *tabPanel*.

Finalment, un cop acabada la implementació del fitxer *UI.r* es procedeix a implementar el fitxer *server.r*. En el cas de la Figura 2.1 i la Figura 2.2 el fitxer *server.r* només conté el codi corresponent a l'únic plot de l'aplicació. Tal com es mostra a continuació, l'histograma s'elabora mitjançant la quantitat d'observacions que l'usuari determina amb la barra lliscant.

```
server <- function(input, output) {
  output$plot1 <- renderPlot({
    set.seed(122)
    histdata <- rnorm(500)
    data <- histdata[seq_len(input$slider)]
    hist(data)
  })
}
```

Així doncs, un cop finalitzada la implementació de les comandes en els fitxers *UI.r* i *server.r* s'executa l'aplicació introduint la comanda `runApp()` a la consola.

3. Estadística Bayesiana

Seguidament s'introduiran conceptes d'estadística Bayesiana per tal d'entendre l'aplicació. Es focalitzarà, sobretot, en els conceptes i tècniques concretes utilitzades durant l'elaboració d'aquesta aplicació.

Els següents punts es basen en els conceptes impartits en un curs estàndard d'estadística Bayesiana **[6]**. Inicialment s'introduirà el concepte d'estadística Bayesiana, conjuntament amb les distribucions a priori, a posteriori i la seva relació. Seguidament es veuran els diferents tipus de distribucions a priori, com es duu a terme la inferència Bayesiana, s'introduiran els contrasts d'hipòtesis Bayesians, es descriurà que es la computació Bayesiana i les seves eines, es descriurà com estimar una regressió lineal simple Bayesiana i es conclourà amb el concepte de punt de canvi i la seva estimació.

3.1 Introducció

Un Model Estadístic, M, és una llista de models de probabilitat indexada per un paràmetre θ, on es sap que θ pertany a un espai de paràmetres Ω, $\theta \in \Omega$, i es pot escriure com $M = \{P(y|\theta); \theta \in \Omega\}$. El Model Estadístic és l'objecte central de l'estadística, és el punt de partida comú de la estadística freqüentista i la Bayesiana abans de bifurcar-se.

El Model Bayesià parteix del model estadístic M, tracta el paràmetre θ com una variable aleatòria i escull una distribució de probabilitat sobre Ω que reflecteix el coneixement a priori sobre θ, aquesta distribució de probabilitat és la priori, $\pi(\theta)$. És a dir, el Model Bayesià és una llista de models de probabilitat ordenada de més a menys creïble segons $\pi(\theta)$.

En l'estadística Bayesiana es fan dos tries subjectives: el model estadístic, que reflecteix el coneixement a priori sobre l'espai mostral i la distribució a priori, que reflecteix el coneixement a priori sobre l'espai paramètric.

Abans d'observar les dades es parteix del model Bayesià $M_B = \{P(y|\theta), \pi(\theta),\ \theta \in \Omega\}$, on $\pi(\theta)$ reflecteix el coneixement o incertesa a priori sobre θ. De la mateixa forma la distribució marginal de les dades, que s'anomena predictiva a priori, $P_\pi(\tilde{y})$, i es calcula com $P_\pi(\tilde{y}) = \int P_\pi(\tilde{y})\pi(\theta)d\theta$ on $\tilde{y}$ representa una dada futura, reflectirà el coneixement o incertesa sobre l'espai mostral abans d'observar les dades.

Per especificar el model Bayesià cal triar els paràmetres de la distribució a priori. Si es té informació a priori sobre θ ens ajudem de la distribució a priori, i si es té sobre l'espai mostral ens ajudem de la predictiva a priori.

Un cop observades les dades l'únic que canvia del model es que s'actualitza la $\pi(\theta)$ per la $\pi(\theta|y)$, on ara la distribució a posteriori reflecteix tot el coneixement sobre el paràmetre un

cop tinguda en compte la informació que han captat les dades, i de la mateixa manera la predictiva a posteriori reflecteix tot el coneixement de les dades futures. La predictiva a posteriori s'utilitza per fer prediccions. En molts casos és més rellevant fer inferència sobre l'espai mostral que no sobre l'espai de paràmetres. Si es vol fer inferència sobre l'espai mostral s'utilitzarà la predictiva a posteriori i si es vol fer inferència sobre l'espai de paràmetres s'utilitzarà la distribució a posteriori.

Per tal de calcular la distribució a posteriori s'utilitza el Teorema de Bayes de la següent forma:

$$\pi(\theta|y) = \frac{P(y,\theta)}{P_\pi(y)} = \frac{P(y|\theta)\pi(\theta)}{P_\pi(y)}.$$

El denominador, $P_\pi(y)$ és la predictiva a priori avaluada a les dades, és a dir, $P_\pi(\tilde{y} = y)$, i és el que pot ser més difícil de calcular ja que s'ha d'integrar: $P\pi(y) = \int_\Omega P(y,\theta)\,d\theta = \int_\Omega P(y|\theta)\pi(\theta)\,d\theta$. Es pot observar que $P_\pi(y)$ és la constant que fa que $P(y|\theta)\pi(\theta)$ integri 1:

$$\int_\Omega \pi(\theta|y)\,d\theta = \int \frac{P(y|\theta)\pi(\theta)}{P_\pi(y)}d\theta = \int \frac{P(y|\theta)\pi(\theta)}{\int P(y|\theta)\pi(\theta)d\theta}d\theta = \frac{1}{\int P(y|\theta)\pi(\theta)d\theta}\int P(y|\theta)\pi(\theta)d\theta = 1,$$

aleshores:

$$\pi(\theta|y) = \frac{P(y|\theta)}{P\pi(y)} = \frac{P(y|\theta)\pi(\theta)}{P\pi(\theta)} \propto P(y|\theta)\pi(\theta) = L_y(\theta)\pi(\theta)$$

on $L_y(\theta)$ és la funció de versemblança.

Moltes vegades no serà necessari calcular $P\pi(y)$, és a dir, no caldrà integrar el denominador ja que pot ser que l'expressió del numerador permeti reconèixer quina distribució segueix $\pi(\theta|y)$.

La distribució a posteriori depèn de les dades nomes a través de la versemblança i per tant respecte el principi de versemblança, que afirma que tota la informació que tenen les dades sobre els paràmetres està a la funció de versemblança. Com més gran és el valor de la versemblança per una θ, més versemblant és que aquest sigui el verdader paràmetre de la població.

3.2 Tipus de distribucions a priori

L'elecció de la distribució a priori és el punt crític de l'estadística Bayesiana. El Bayesià a més a més de triar el model estadístic també ha d'escollir una distribució a priori sobre l'espai de paràmetres Ω que reflecteixi o capturi el coneixement, o incertesa, sobre θ, i s'ha de fer abans d'observar les dades.

La distribució a priori es construeix utilitzant la informació d'estudis anteriors i/o amb l'opinió consensuada dels experts. En el cas de no tenir estudis previs, de no poder consensuar l'opinió dels experts o de voler que les dades parlin soles sempre es pot utilitzar distribucions no informatives.

Hi ha diferents tipus de distribucions a priori, les informatives que es classifiquen en conjugades i no conjugades, i les a priori no informatives que es classifiquen en la priori plana, la priori de Jeffreys i el cas límit de les conjugades.

3.2.1 Priori informatives

A partir d'informació prèvia, aquesta es vol canalitzar utilitzant una distribució de probabilitat, això vol dir que s'han de triar els paràmetres d'aquesta distribució. Es poden triar per assaig i error ajustant la gràfica de la distribució visualment de manera que es doni més densitat a aquells valors que a priori es consideren més probables, o també es poden triar igualant moments, quartils, etc.

Si la informació que es té a priori és sobre l'espai mostral, aleshores es trien els paràmetres de la distribució a priori dibuixant la predictiva a priori.

- **Conjugades**

Una distribució a priori és la conjugada d'un model estadístic si la distribució a posteriori és de la mateixa família que la distribució a priori. Aleshores la distribució a posteriori s'obté actualitzant els paràmetres de la distribució a priori.

Les demostracions d'algunes distribucions a posteriori són:

- **Binomial**

En el model $Binomial$, $M = \{Binomial(\theta, m), \theta \in (0,1)\}$, s'escull com a priori una distribució $Beta$, $\pi(\theta) = Beta(a, b)$[1], ja que θ pren valors entre 0 i 1. Aleshores es calcula la distribució a posteriori com:

[1] En les distribucions, el Bayesià utilitza les lletres gregues per determinar variables aleatòries i les lletres llatines per determinar valors.

$$\pi(\theta|y) = \frac{P(y|\theta)\pi(\theta)}{P\pi(y)} \propto P(y|\theta)\pi(\theta) = L_y(\theta)\pi(\theta) =$$
$$= \binom{n}{y}\theta^y(1-\theta)^{n-y}\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1} \propto \theta^{a+y-1}(1-\theta)^{b+n-y-1}$$
$$\Rightarrow \pi(\theta|y) = Beta(a+y, b+n-y).$$

- **Poisson**

En el model $Poisson$, $M = \{Poisson(\theta), \theta \in \mathbb{R}^+)\}$, s'escull com a priori una distribució $Gamma$, $\pi(\theta) = Gamma(a,b)$, ja que θ pren valors positius. Aleshores es calcula la distribució a posteriori com:

$$\pi(\theta|y) = \frac{P(y|\theta)\pi(\theta)}{P\pi(y)} \propto L_y(\theta)\pi(\theta) = e^{-n\theta}\frac{\theta^{\sum y_i}}{\prod y_i!}\frac{b^a\theta^{(a-1)}e^{-b\theta}}{\Gamma(a)} \propto e^{-n\theta}\theta^{\sum y_i}\theta^{a-1}e^{-b\theta}$$
$$= e^{-\theta(n+b)}\theta^{\sum y_i+a-1} \Rightarrow \pi(\theta|y) = Gamma\left(a+\sum_{i=1}^{n} y_i\,, b+n\right).$$

- **No Conjugades**

El ventall de distribucions que es pot utilitzar es tant gran com es vulgui. Com que es té més opcions és més difícil escollir. S'haurà de calcular quina és la distribució a posteriori integrant el denominador de l'expressió:

$$\pi(\theta|y) = \frac{L_y(\theta)\pi(\theta)}{\int L_y(\theta)\pi(\theta)d\theta},$$

o bé utilitzar mètodes de computació Bayesiana per aproximar la distribució a posteriori utilitzant simulacions.

3.2.2 Priori no informatives

Es tracta de que $\pi(\theta)$ interfereixi el mínim amb les dades. Existeixen varies opcions, a la pràctica es pot fer una anàlisi de sensibilitat que consisteix a utilitzar diferents distribucions a priori no informatives i avaluar com canvia la distribució a posteriori.

- **Priori plana o de Laplace**

Consisteix en triar $\pi(\theta)$ constant, $\pi(\theta) = k$. Així es té:

$$\pi(\theta|y) = \frac{L_y(\theta)\,k}{\int L_y(\theta)\,k\,d\theta} \propto L_y(\theta).$$

- **Priori de Jeffreys**

Es calcula com $\pi(\theta) \propto \sqrt{|I(\theta)|}$ on $I(\theta)$ és la informació de Fisher, que s'interpreta com la quantitat d'informació que té el model en θ.

- **Cas límit de les conjugades**

Partint de la distribució conjugada es trien els paràmetres que facin que la variància del paràmetre tendeixi a infinit. A la pràctica moltes vegades s'escullen els paràmetres de la distribució conjugada que fan la variància molt gran.

3.3 Inferència Bayesiana

Fer inferència significa que a partir de la mostra es vol saber com és la població. El millor estimador és la distribució a posteriori, $\pi(\theta|y)$, perquè té tota la informació que es coneix del paràmetre, ja que té en compte la informació a priori i la informació de les dades.

Per fer inferència Bayesiana es pot obtenir una estimació puntual de la distribució a posteriori o, una estimació per interval d'aquesta.

3.3.1 Estimació puntual

Com a estimador puntual del paràmetre θ, que és el paràmetre del model estadístic, s'utilitza qualsevol mesura de localització de la distribució a posteriori, per exemple:

- L'esperança: $\hat{\theta}_{ep} = E(\theta|y) = \int_{\Omega} \theta\pi(\theta|y)d\theta$.
- La moda: $\hat{\theta}_{modp} = Moda(\theta|y)$.

3.3.2 Estimació per interval

Es defineix un interval de credibilitat a posteriori amb probabilitat p, C_p de θ, a qualsevol conjunt de Ω tal que $P(\theta \in C_p|y) = p$, és a dir, $P(\theta \in C_p|y) = \int_{C_p} \pi(\theta|y)d\theta = p$.

Es poden calcular els intervals de credibilitat de moltes formes, però a la pràctica només se n'utilitzen dues: els intervals de credibilitat de màxima densitat i els intervals centrals de credibilitat.

En aquest treball s'utilitzaran els intervals centrals de credibilitat, que són els més utilitzats i es basen en els percentils. Són invariants davant de reparametritzacions i fàcils de calcular a partir de simulacions de la distribució a posteriori. L'únic que s'ha de fer per construir l'interval és ordenar les simulacions per obtenir els percentils $\frac{1-p}{2}$ i $1 - \frac{1-p}{2}$.

Es poden calcular intervals de credibilitat de qualsevol distribució, tant de les distribucions a priori i posteriori com de les predictives.

3.4 Contrast de dues hipòtesis

El contrast de dues hipòtesis s'utilitza quan es vol decidir si $\theta \in \Omega_1$ o $\theta \in \Omega_2$, on $\Omega \in \{\Omega_1 \cup \Omega_2\}$ i $\Omega_1 \cap \Omega_2 = \emptyset$.

$$\begin{cases} H_1: \theta \in \Omega_1 \\ H_2: \theta \in \Omega_2 \end{cases} \textit{és equivalent a } \begin{cases} H_1: y|\theta \sim P_1(y|\theta), P_1(y|\theta) \in \{P(y|\theta), \theta \in \Omega_1\} \\ H_2: y|\theta \sim P_2(y|\theta), P_2(y|\theta) \in \{P(y|\theta), \theta \in \Omega_2\} \end{cases}$$

Triar una hipòtesi és triar un submodel. Per un Bayesià tot el que es pot fer amb dades també es pot fer sense, i per tant es poden fer contrasts d'hipòtesis sense dades. Un cop es tenen les dades es calculen les probabilitats a posteriori de les hipòtesis utilitzant la distribució a posteriori (en cas de no tenir dades es substitueix la distribució a posteriori per la distribució a priori):

$$\begin{cases} P(H_1|y) = \int_{\Omega_1} \pi(\theta|y) d\theta, \\ P(H_2|y) = \int_{\Omega_2} \pi(\theta|y) d\theta \,. \end{cases}$$

D'aquesta forma, un cop obtingudes les probabilitats a posteriori de les hipòtesis es tria la hipòtesis amb una probabilitat més alta.

Les hipòtesis són simètriques, no hi ha hipòtesi nul·la i alternativa. És fàcil obtenir les probabilitats a posteriori de les hipòtesis via simulació. Així el percentatge de simulacions que pertanyen a Ω_1 serà una aproximació de $P(H_1|y)$ i el percentatge de simulacions que pertanyen a Ω_2 serà una aproximació de $P(H_2|y)$. Les probabilitats de les hipòtesis sempre sumen 1 i s'escull la hipòtesi que tingui la probabilitat més gran.

El contrast d'hipòtesis Bayesià es pot generalitzar al cas de més de dues hipòtesis, hi ha situacions molt més versàtils, com per exemple que l'espai de paràmetres de les hipòtesis no siguin disjunts, és a dir, que les interseccions entre els espais de paràmetres siguin el buit.

3.5 Computació Bayesiana

La computació Bayesiana fa referència a tots els procediments que aproximen la distribució a posteriori, generalment a través de la simulació. La computació Bayesiana no és estadística, sinó que són eines que utilitza l'estadística Bayesiana.

Tota la inferència Bayesiana es basa en obtenir la distribució a posteriori i les predictives a posteriori. Moltes vegades aquestes distribucions no es poden obtenir de forma tancada, és a dir, tenir de forma explícita la seva funció de densitat, ja que les integrals són difícils de calcular o simplement no tenen primitiva. Per calcular $\pi(\theta = (\theta_1, \dots, \theta_2)|y)$ s'ha de resoldre:

$$\pi(\theta = (\theta_1, \dots, \theta_p)|y) = \frac{L_y(\theta_1, \dots, \theta_p)\pi(\theta_1, \dots, \theta_p)}{\int_{\Omega_1} \dots \int_{\Omega_p} L_y(\theta_1, \dots, \theta_p)\pi(\theta_1, \dots, \theta_p) d\theta_p \dots d\theta_1}.$$

Si a partir del numerador es pot identificar de quina distribució es tracta, aleshores no és necessari calcular les integrals, tal com passa en el cas de les distribucions conjugades.

Si $\theta = (\theta_1, \dots, \theta_p) \in \mathbb{R}^p$ i p és petit, es pot aproximar el denominador numèricament, aproximant la integral per sumes. Quan p es gran i les priori són no informatives no és viable aquesta aproximació perquè es trobarien problemes greus de precisió. En aquest cas és on entra el paper de la computació Bayesiana i les simulacions.

3.5.1 Simulacions MCMC

Els mètodes MCMC (Markov Chain Monte Carlo) són els més utilitzats. Aquests mètodes permeten simular la distribució a posteriori a base de dividir el problema en problemes més petits. Aquests algoritmes basats en Cadenes de Markov garanteixen que la distribució estacionària és la distribució a posteriori. La dificultat d'aquests mètodes és decidir quan ha convergit la cadena, és a dir, quan està simulant de la distribució a posteriori.

Hi ha diversos algoritmes MCMC, els més coneguts són:

- Gibbs Sampling.
- *Metropolis – Hasting*, que es pot entendre com una generalització del *Gibbs Sampling.*

En la implementació de l'aplicació, quan ha sigut necessari simular de la distribució a posteriori, s'ha utilitzat l'algoritme de *Gibbs Sampling.*

- ***Gibbs Sampling***

Es suposa que θ té dimensió p, $\theta = (\theta_1, \dots, \theta_p)$, aleshores l'algoritme consisteix en escriure les distribucions condicionals $\pi(\theta_1|\theta_2, \dots, \theta_p, y), \dots, \pi(\theta_p|\theta_2, \dots, \theta_{p-1}, y)$. Seguidament s'escullen uns valors inicials $\theta_1^{(0)}, \dots, \theta_p^{(0)}$ arbitraris i per $m = 1, \dots, k$ es repeteix:

1- Es simula $\theta_1^{(m)}$ de $\pi(\theta_1|\theta_2^{(m-1)}, \dots, \theta_p^{(m-1)}, y)$

2- Es simula $\theta_2^{(m)}$ de $\pi(\theta_2|\theta_1^{(m)}, \theta_3^{(m-1)}, \dots, \theta_p^{(m-1)}, y)$

⋮

p- Es simula de $\theta_p^{(m)}$ de $\pi(\theta_p|\theta_1^{(m)}, \dots, \theta_{p-1}^{(m)}, y)$

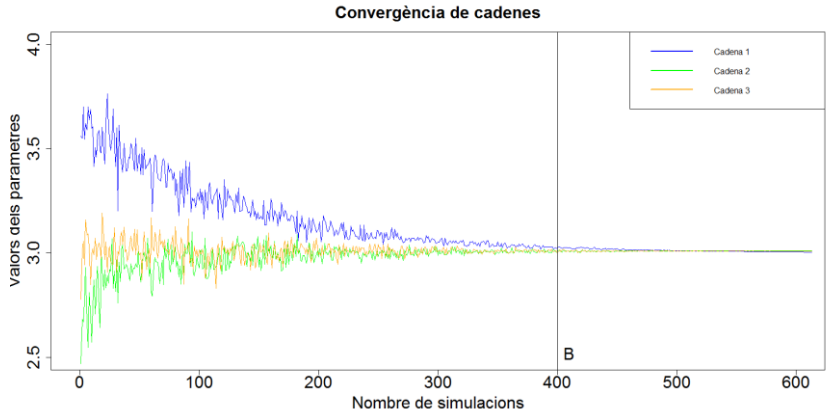
Després d'un període d'escalfament, anomenat B, si la cadena ha convergit, les simulacions $\theta_1^{(m)}, \dots, \theta_p^{(m)}$ $m = B+1, \dots, M$ són simulacions de la distribució a posteriori $\pi(\theta_1, \dots, \theta_p | y)$.

M	θ_1	...	θ_p
1	$\theta_1^{(1)}$	...	$\theta_p^{(1)}$
...	...	...	...
B	$\theta_1^{(B)}$	...	$\theta_p^{(B)}$
B+1	$\theta_1^{(B+1)}$	...	$\theta_p^{(B+1)}$
...	...	...	...
M	$\theta_1^{(M)}$	...	$\theta_p^{(M)}$

En l'elaboració de l'aplicació s'utilitzarà l'algoritme *Gibbs Sampling* per tal d'estimar la distribució a posteriori dels paràmetres del model estadístic $Normal$, del model de regressió lineal simple i del model $Binomial$ amb presència de punt de canvi..

- ***Avaluació de la convergència de les cadenes***

Una cadena és una seqüència de simulacions. L'objectiu d'avaluar la convergència de la cadena o cadenes és decidir quan han convergit, és a dir, escollir el punt B a partir del qual la cadena estaciona en un valor. Les simulacions anteriors al punt B s'anomenen simulacions d'escalfament.



Gràfic 3.1 Convergència de les cadenes a partir del punt *B*

No hi ha cap fórmula per escollir B, l'estratègia que es recomana és:

- Fer córrer l'algoritme diferents vegades a partir de diferents i dispersos valors inicials.

- Fer una gràfica de cadascuna de les cadenes per veure en quin moment es solapen, quan les cadenes s'han barrejat s'assumeix que s'està simulant de la distribució a posteriori. Tal com es pot veure al Gràfic 3.1.

Existeixen mesures complementàries a la inspecció visual per determinar la convergència de les cadenes. La més utilitzada és *R-hat*, $(\hat{R})$, proposat per Gelman i Rubin.

Per calcular el *R-hat*, $(\hat{R})$ s'ha de dividir en blocs el nombre total de simulacions, es denota $C_i^{(j)}$ com les simulacions de la cadena i en el bloc j, aleshores $C_j = \{C_1^j, C_2^j, \dots\}$ són les simulacions totals en el bloc j. Aleshores *R-hat* és la variància de les simulacions totals en el bloc j dividit per la mitjana de les variàncies de cada cadena en el bloc j, és a dir:

$$\hat{R}_j = \frac{Var(C_j)}{\left(\frac{Var\left(C_1^{(j)}\right) + Var\left(C_2^{(j)}\right) + \cdots + Var\left(C_I^{(j)}\right)}{I}\right)},$$

on I és el nombre total de cadenes. Aleshores quan $\hat{R}_j \cong 1$ significa que les cadenes en aquell bloc han convergit, en canvi quan $\hat{R}_j \gg 1$ significa que les cadenes encara no han convergit.

3.6 Regressió lineal simple

La regressió lineal simple estudia com varia una variable aleatòria Y en funció d'una altra variable X que es suposa coneguda. La regressió lineal simple assumeix que la relació entre X i Y és lineal.

El model Bayesià més habitual per la regressió lineal simple és:

$$y_1, \dots, y_n | \beta_0, \sigma, x_i \sim \prod_{i=1}^{n} Normal(\beta_0 + \beta_1 x_i, \sigma),$$

$$\beta_0 \sim Normal(m_0, s_0),$$

$$\beta_1 \sim Normal(m_1, s_1),$$

$$\frac{1}{\sigma^2} \sim Gamma(a, b),$$

on $\frac{1}{\sigma^2}$ és la precisió. Aleshores $\theta = (\beta_0, \beta, \sigma)$.

Per estimar el model lineal simple s'utilitzen algoritmes de MCMC. El programa *WinBugs* o les llibreries específiques d'estadística Bayesiana en *R* permeten estimar el model a posteriori.

3.7 Punt de canvi

A partir d'unes dades ordenades (per exemple cronològic), estimar el punt de canvi (r) significa estimar quines observacions pertanyen a un model de probabilitat i quines a un altre. És a dir, en la seqüencia de dades les primeres r observacions provenen d'un model de probabilitat i les següents observacions provenen d'un altre. L'estimació del punt de canvi es duu a terme mitjançant l'algoritme del *Gibbs Sampling*.

A partir del model estadístic $M = \{P(y|\theta); \theta \in \Omega\}$ amb $\theta = (\theta_1, \theta_2)$ l'objectiu és determinar quines observacions provenen del model amb θ_1, quines del model amb θ_2, i sobretot determinar en quin moment r s'ha produït el canvi. Es pot escriure el model estadístic com:

$$y_i, \dots, y_n|\theta_1, \theta_2, r \sim \prod_{i=1}^{r} P(y_i|\theta_1) \prod_{i=1+r}^{n} P(y_i|\theta_2).$$

En la aplicació s'estimarà el punt de canvi a partir d'una mostra provinent de $Binomial(\theta_i, m_i)$. Per l'estimació del punt de canvi s'haurà de trobar les distribucions condicionades dels paràmetres a les dades. Els càlculs d'aquestes distribucions es troben més endavant.

4. StatClip

StatClip és una aplicació *Shiny* creada per l'Eduard Serrahima de Cambra pel seu treball de fi de grau defensat el juny del 2015 **[8]**. L'Eduard va estudiar el Grau en Enginyeria en Tecnologies Industrials a l'Escola Tècnica Superior d'Enginyeria Industrial de Barcelona (ETSEIB) i va tenir de tutor el professor Lluís Marco (que també supervisa aquest treball).

La motivació del treball de l'Eduard era reforçar el paper de l'estadística en l'educació, ja que aquesta juga un paper molt important en l'actualitat pel fet d'haver d'analitzar el gran creixement de dades disponibles. En la docència primària i secundària, l'estadística s'explica mínimament i sense el suport d'unes eines adequades.

Arrel d'aquesta motivació, va sorgir l'objectiu de crear un suport per la docència de l'estadística. Un suport on els alumnes es dediquin a interioritzar els conceptes estadístics i a interpretar els gràfics i resultats, no es desitja un suport en el que hagin d'estar més pendents de com s'elaboren els càlculs que de la seva interpretació.

Els objectius del seu projecte eren:

- Realitzar una recerca exhaustiva d'aplicacions ja existents (incloent tant programari comercial com programari lliure, així com llenguatges de programació). És molt important entendre el que ja existeix i intentar saber què es necessita. També ajuda a trobar la inspiració de cara al disseny *StatClip*.
- Decidir a qui va adreçada l'aplicació i analitzar exhaustivament les seves necessitats.
- Definir la llista de característiques del programa.
- Dissenyar teòricament totes les característiques i funcionalitats de *StatClip*, així com la interfície d'usuari.
- Triar una eina computacional sobre la qual construir l'aplicació.
- Dissenyar un prototip de *StatClip*, fent servir l'eina computacional seleccionada. Més concretament, construir l'arquitectura interna de l'aplicació, i implementar les seves funcionalitats, sempre tenint en compte de fer que *StatClip* pugui ser ampliada.
- Comprovar i validar el disseny de l'aplicació, fent servir tècniques de disseny emocional.

Així doncs, després d'una recerca, es va decidir que la millor forma d'implementar *StatClip* era utilitzant el paquet *Shiny* del programa *R* ja que en *R* s'hi troben totes les eines estadístiques necessàries, *Shiny* permet utilitzar aplicacions sense la necessitat de saber codi *R*, és programari lliure i permet la incorporació de l'aplicació en una pàgina *web*.

Els elements finals de *StatClip*, amb una implementació complerta, van ser:

- *Data*: secció per seleccionar les dades a analitzar.
    - *Load Data Set*: permet seleccionar alguna base de dades d'exemple o carregar una pròpia base de dades mitjançant "copia i enganxa".
    - *Create Simulated Data*: permet crear una taula amb valors aleatoris de qualsevol model de probabilitat.
- *Graphs*: secció per analitzar gràficament les dades introduïdes.
    - *Histogram*: elabora un histograma i permet determinar diferents opcions del gràfic, ja sigui en càlcul o aparença.
    - *Scatterplot*: elabora un gràfic de dispersió amb les variables *X*, *Y* i una possible variable d'estratificació. Permet determinar diferents opcions d'aparença.
    - *Bubble Plot:* elabora un gràfic de bombolles. És com un gràfic de dispersió però incorpora una variable on la mida de les bombolles van en funció d'aquesta.
    - *Maps:* elabora un gràfic de tipus mapa, és a dir, mitjançant una variable de latitud, una de longitud i possiblement una variable d'estratificació es representen els punts en un mapa mundi.
- *Computations:* secció que conté diferents càlculs.
    - *Probabilities:* mitjançant un model de probabilitat dóna l'àrea de la funció de densitat segons l'interval indicat.
    - *Descriptive Statistics:* dóna una taula resum de les estimacions puntuals per les diferents variables seleccionades.

Per a una implementació estructuralment clara es va seguir el diagrama de la Figura 4.1. D'aquesta forma es poden afegir seccions a l'aplicació sense una gran alteració del contingut existent i la reestructuració dels arxius.

Així doncs, es va dissenyar des de zero una aplicació per a l'aprenentatge de l'anàlisi estadístic de dades. El resultat va ser un programa completament funcional amb un ventall útil de funcionalitats implementades. El que és més important, però, és que tota l'arquitectura i estructura del *software* va ser dissenyada per facilitar l'addició de funcionalitats a *StatClip*. D'aquesta forma, *StatClip* pot continuar creixent i evolucionant per convertir-se en una eina educacional completa i útil.

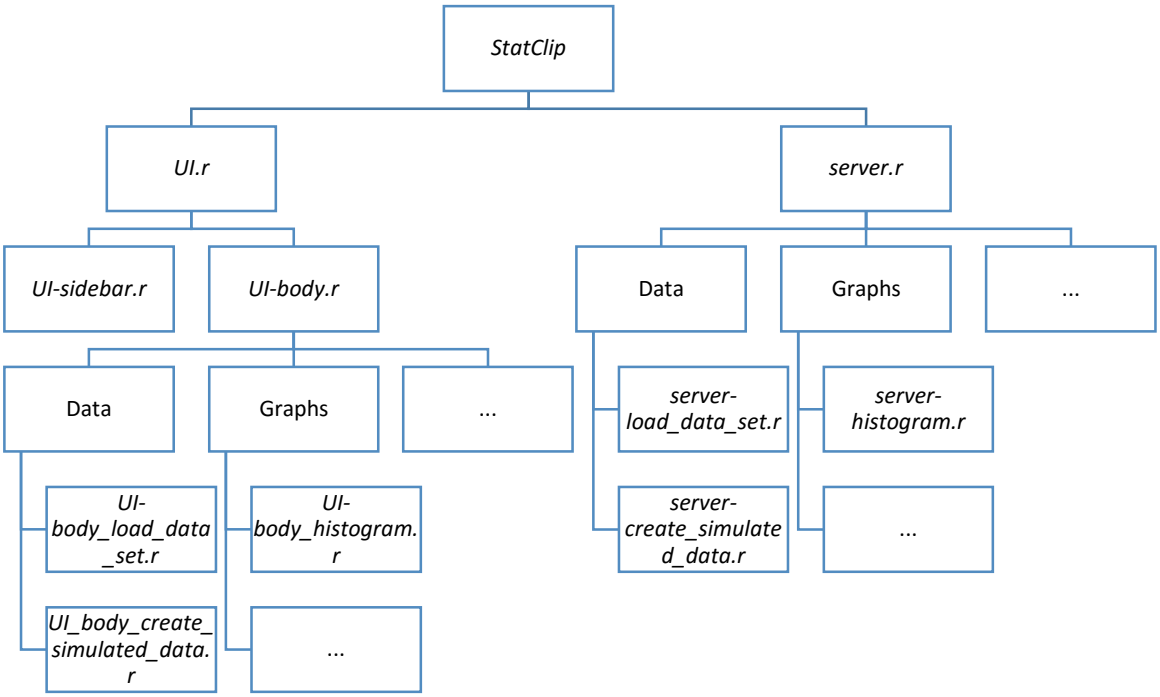


Figura 4.1 Estructura dels arxius per la implementació de *StatClip*

5. Aplicacions existents d'estadística Bayesiana

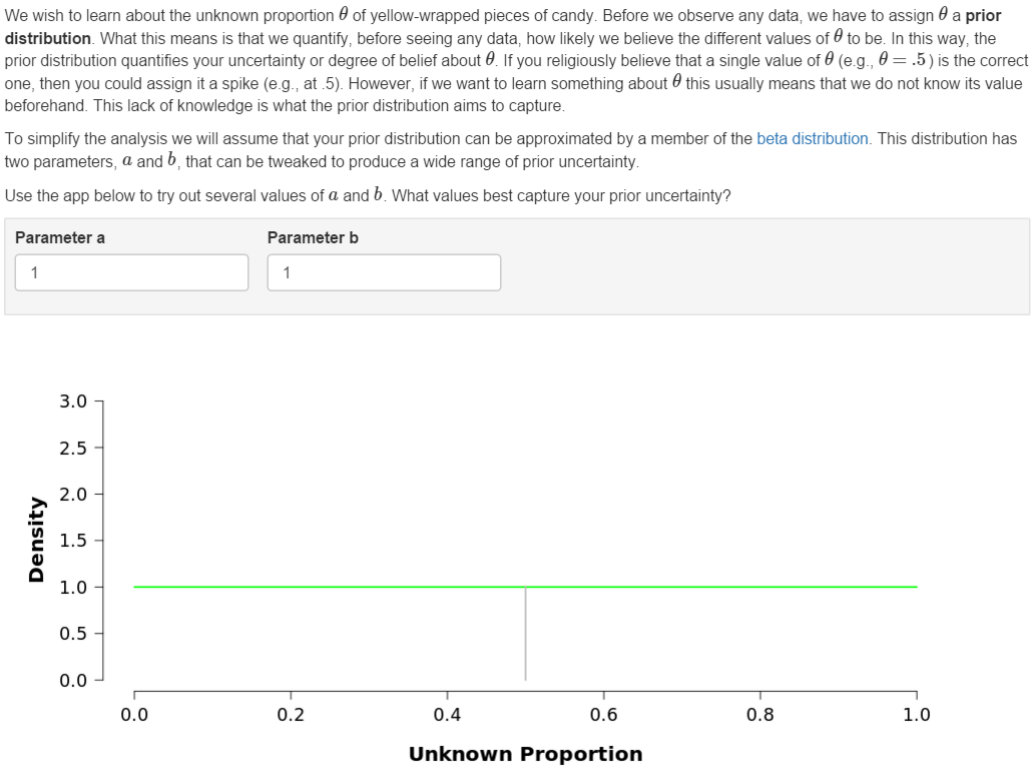
Abans de començar a dissenyar l'aplicació es fa una recerca d'altres aplicacions existents sobre estadística Bayesiana per tal d'agafar idees i innovar. Cal esmentar que existeix un programa anomenat *WinBugs*, que és una eina potent d'estadística Bayesiana, però no és senzill ni intuïtiu i per tant no seria adequat per un curs introductori d'estadística Bayesiana. En aquesta secció es fa una cerca de les eines senzilles i intuïtives que encaixen amb la idea del projecte.

S'han trobat un total de 4 aplicacions relacionades amb l'estadística Bayesiana, totes elles amb una interfície d'usuari simple.

- ***Aplicació 1***

En la Imatge 5.1 es mostra la imatge de l'aplicació *A First Lesson in Bayesian* Inference **[9]**. Aquesta està implementada en un arxiu *Markdown* que conté codi Shiny. L'aplicació conté molt de text explicatiu. És un exemple específic ja que està focalitzat en resoldre un problema. L'aplicació explica el problema i fa escollir diferents valors dels paràmetres d'una distribució $Beta$ per observar com canvia la funció de densitat. Seguidament es proposen varies preguntes per després explicar el concepte de la distribució a posteriori i observar com canvia la seva distribució alterant els paràmetres i les dades. Finalment s'explica el concepte del Factor de Bayes i es representa gràficament.

Es conclou que aquest arxiu conté massa text explicatiu i només és útil per la distribució a priori $Beta$. A favor cal destacar que el gràfic de densitat de la priori i posteriori són clars.

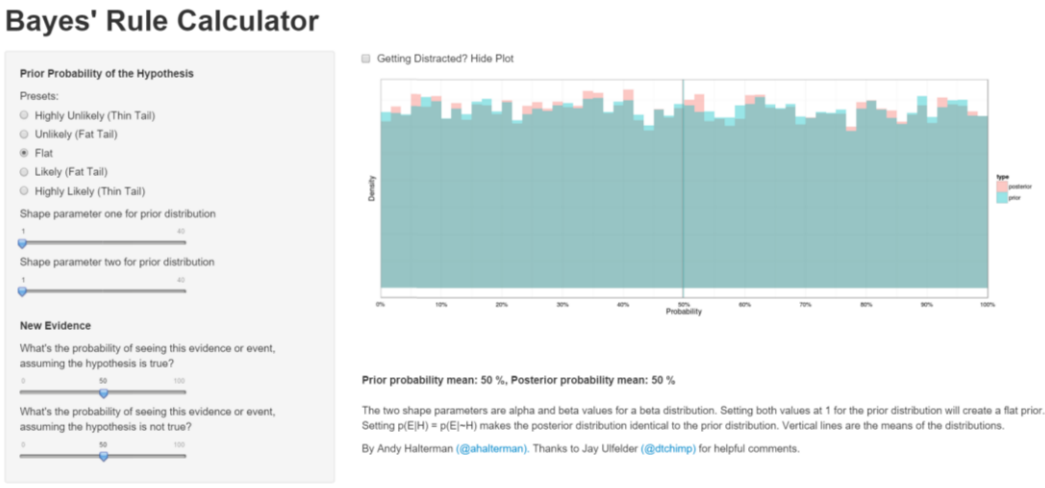


Imatge 5.1 Interfície d'usuari de l'aplicació *A First Lesson in Bayesian Inference*

- ***Aplicació 2***

La Imatge 5.2 mostra la interfície d'usuari de l'aplicació *Bayes' Rule calculator* **[10]**. Aquesta és una aplicació Shiny que està penjada al servidor de *Shinyapps*. L'aplicació parteix d'una distribució $Beta$, on s'escullen els seus paràmetres a partir d'opcions referents a la probabilitat prèvia de les hipòtesis, com altament probable (cua estreta), improbable (cua grossa), plana, probable i molt probable. Altrament també es poden escollir els paràmetres a partir de les *sliders* (barres lliscants). Seguidament manipulant les probabilitats de la hipòtesi assumint que es certa o falsa s'observa com canvia el gràfic amb la distribució a priori i posteriori. També dóna les mitjanes a priori i posteriori de les distribucions.

Es conclou que aquesta aplicació només és útil per la distribució $Beta$ i el fet de donar les probabilitats condicionades a verdader o fals es de difícil comprensió.

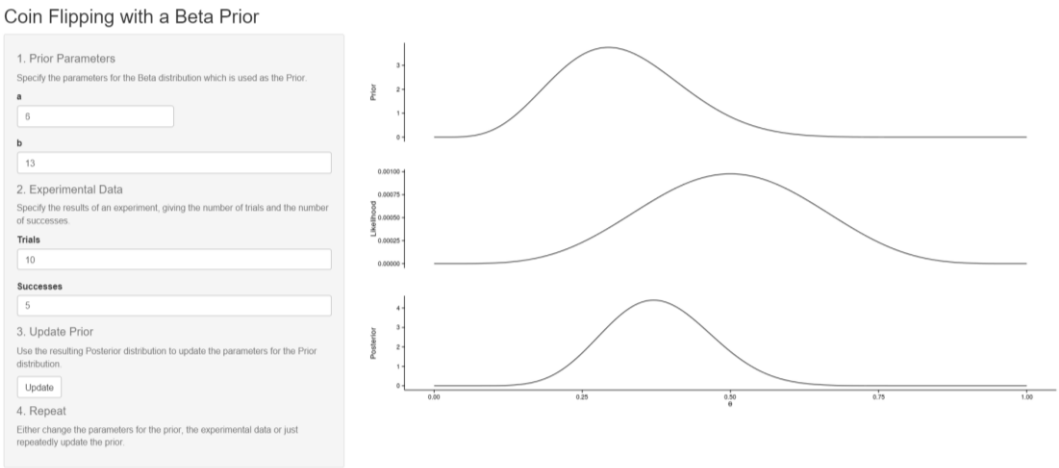


Imatge 5.2 Interfície d'usuari de l'aplicació *Bayes' Rule Calculator*

- ***Aplicació 3***

La Imatge 5.3 mostra la interfície d'usuari de l'aplicació *Coin Flipping with a Beta Prior* **[11]**, aquesta és una aplicació *Shiny* que treballa amb una distribució $Beta$. A la dreta de l'aplicació es troben les opcions i a l'esquerra els gràfics de la distribució a priori, la versemblança i la distribució a posteriori. Les primeres opcions fan escollir els paràmetres de la distribució a priori, seguidament s'introdueixen les dades, és a dir, el nombre de proves i el nombre d'èxits. Seguidament, mitjançant un botó *update* s'actualitza la distribució a priori amb els resultats de la distribució a posteriori.

Es conclou que aquesta aplicació només és útil per la distribució $Beta$. La presentació de l'aplicació i la disposició dels outputs és molt bona.

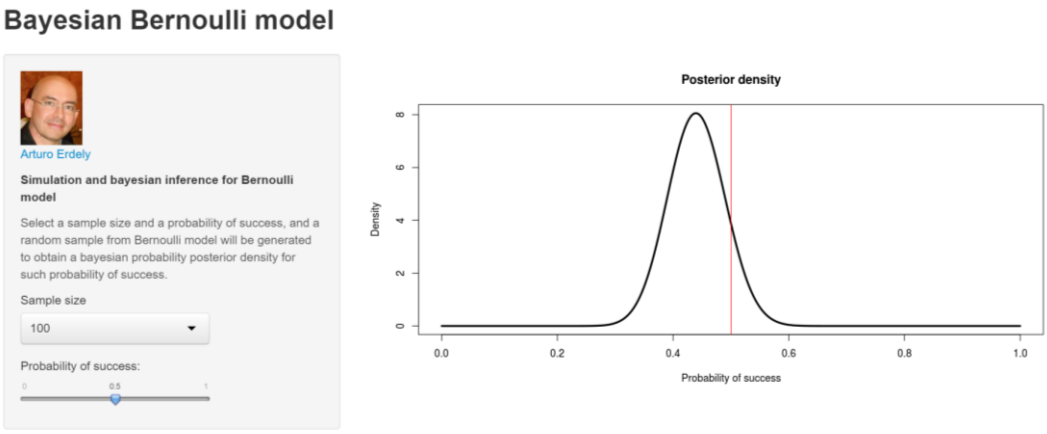


Imatge 5.3 Interfície d'usuari de l'aplicació *Coin Flipping with a Beta Prior*

- ***Aplicació 4***

La Imatge 5.4 mostra la interfície d'usuari de l'aplicació *Bayesian Bernoulli* model penjada al servidor de *Shinyapps* **[12].** Aquesta aplicació tracta la distribució $Bernoulli$. Hi ha un petit text explicatiu que indica que s'ha de seleccionar la mida de la mostra i la probabilitat d'èxit per generar una mostra aleatòria i obtenir la funció de densitat a posteriori. Només inclou un gràfic, el de la distribució a posteriori, amb una línia vertical vermella que indica la probabilitat a priori del succés.

Es conclou que aquesta aplicació nomes és útil per la distribució $Bernoulli$ i que la presentació de la aplicació és molt simple.



Imatge 5.4 Interfície d'usuari de l'aplicació *Bayesian Bernoulli model*

Finalment, després d'analitzar les aplicacions Bayesianes de *Shiny* trobades, la que més utilitat s'ha trobat que té és la de *Coin Flipping with a Beta Prior,* Imatge 5.3*,* pels inputs i outputs que conté així com la disposició a la pàgina d'aquests.

6. BayesClip

L'elaboració de l'aplicació *BayesClip*, sobretot la implementació del codi, s'ha emportat la major part del temps dedicat en aquest projecte. El fet d'haver d'adaptar-se inicialment a una estructura de codi que no ha iniciat un mateix suposa una dificultat afegida.

Per decidir el disseny de l'aplicació, és a dir, l'organització dels seus elements en la interfície d'usuari, s'han confeccionat figures mitjançant el programa *Balsamiq Mockups 3* **[13]**. Aquest programa ha permès elaborar tota mena de dissenys per poder prendre les decisions pertinents.

BayesClip ha de seguir l'estructura interna i l'aparença de *StatClip*, és per això que l'aplicació té la interfície d'usuari bàsica que s'aprecia en la Figura 6.1.

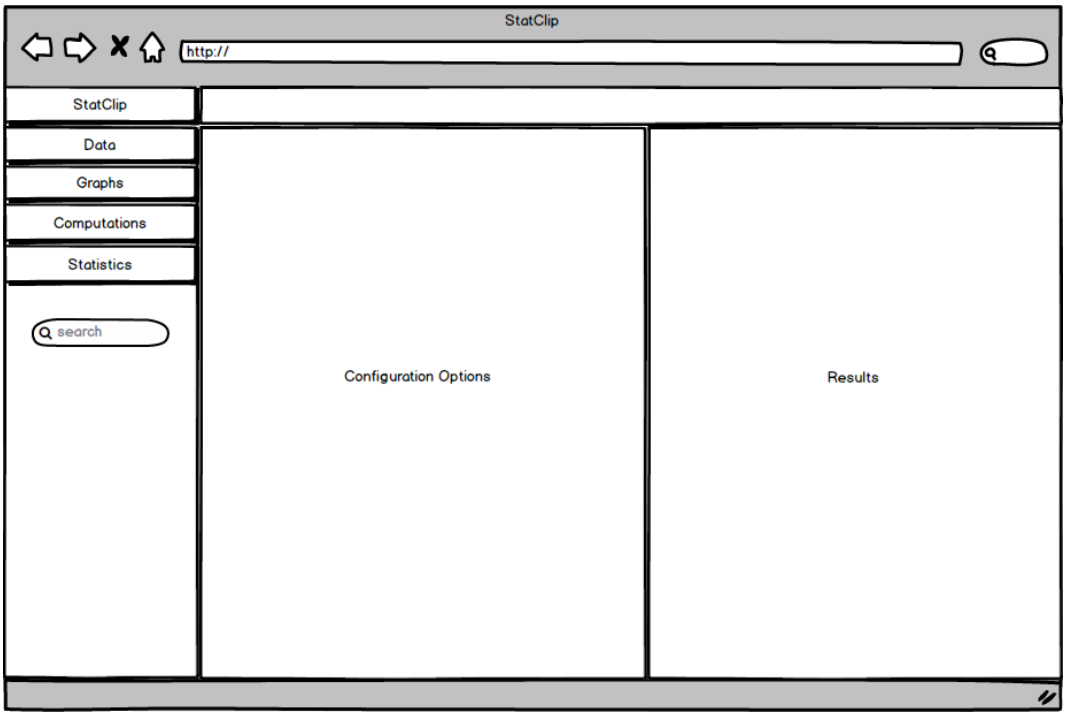


Figura 6.1 Aparença bàsica de l'aplicació *BayesClip*

Cal esmentar que els títols informatius de les distribucions s'implementaran amb codi HTML per tal d'incorporar lletres gregues. Utilitzar codi HTML en una aplicació *Shiny* és molt senzill. Dins de la funció corresponent, en aquest cas *h3* per tal de representar un títol, s'incorpora la funció HTML() i dins d'aquesta s'hi escriu el codi directament en format de pàgina web. Per exemple, per escriure $\theta_1 \sim Beta(\alpha, \beta)$ s'utilitza:

```
h3(HTML("&theta;<sub>1</sub> ~ Beta(a, b)"))
```

L'estructura interna dels fitxers de codi de *BayesClip* és jeràrquica, com la de *StatClip*. La implementació de la interfície d'usuari s'ha dividit en diferents arxius, un per a cada apartat. L'arxiu *UI-body.R* conté la crida dels arxius de cada apartat i aquest, a la vegada, és cridat pel fitxer *UI.r* tal com mostra la Figura 6.2. Els fitxers *server.r* també han estat dividits d'aquesta forma. A més, alguns fitxers *server.r* necessiten cridar altres fitxers en format *R* que contenen funcions auxiliars, tal com mostra la Figura 6.3.

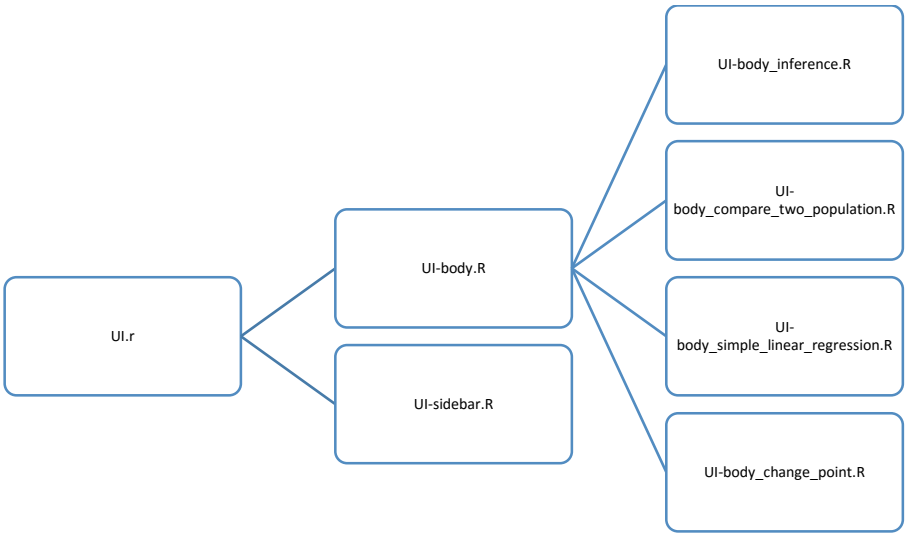


Figura 6.2 Estructura dels arxius *UI.r*

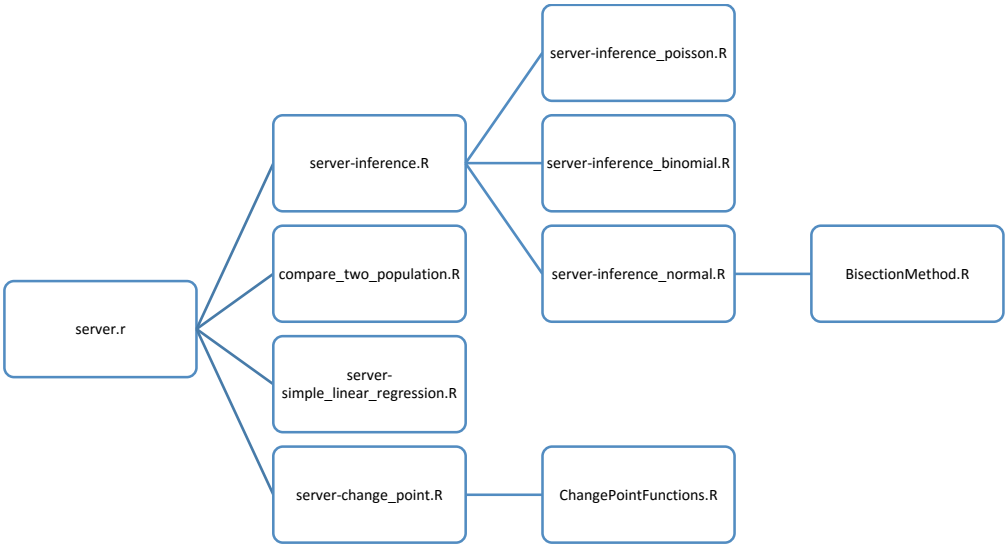


Figura 6.3 Estructura dels arxius *server.r*

6.1 Càrrega de dades

La part *Data* implementada en l'aplicació *StatClip* permet treballar amb dues bases de dades, *Iris data set* i *Mtcrs data set,* que es troben per defecte. També és possible generar nombres aleatoris de qualsevol distribució o, altrament, enganxar les dades necessàries directament a l'aplicació.

Ja que l'aplicació serà penjada en un servidor *web*, la funció que permet enganxar les dades directament a l'aplicació, la funció *Clipboard,* presenta algun tipus de problema i no efectua bé la lectura de dades a l'hora de treballar en el servidor.

Així doncs, en lloc de que l'usuari enganxi les dades, s'opta per implementar la lectura de dades mitjançant la càrrega d'un arxiu en format *.txt* o *.csv*.

6.1.1 Aparença

L'aparença de la secció *Data*, concretament l'apartat de *Load Data Set*, de l'aplicació *StatClip* és la que s'observa en la Figura 6.4. En aquesta es pot observar el botó que permet enganxar les dades.

Per modificar la forma de càrrega les dades es busca en *Shiny-Gallery* i es troba un disseny per carregar arxius a una aplicació web **[14]**. En aquest disseny l'usuari pot indicar si la base de dades té o no capçalera, si les dades es troben separades per coma (,), punt i coma (;) o tabulació (\t) i si les dades es troben entre cometes simples (‘), cometes dobles (“) o sense cometes ( ). Així doncs s'adapta estèticament a l'aplicació de tal forma com mostra la Figura 6.5.

6.1.1 Implementació

Es modifica el codi existent del fitxer *server-load_data_set.R*. Es substitueix la funció *Clipboard,* la responsable de poder enganxar les dades directament, per la funció *read.csv(),* que permet llegir fitxers*,* amb les seves corresponents especificacions, que s'obtindran dels *inputs* implementats a continuació.

Seguidament s'implementen els inputs en el fitxer *UI-body_load_data_set.R*. S'implementa un botó que obre un navegador per tal de seleccionar el fitxer de dades, un requadre per indicar si les dades tenen capçalera, i les opcions dels separadors determinades anteriorment.

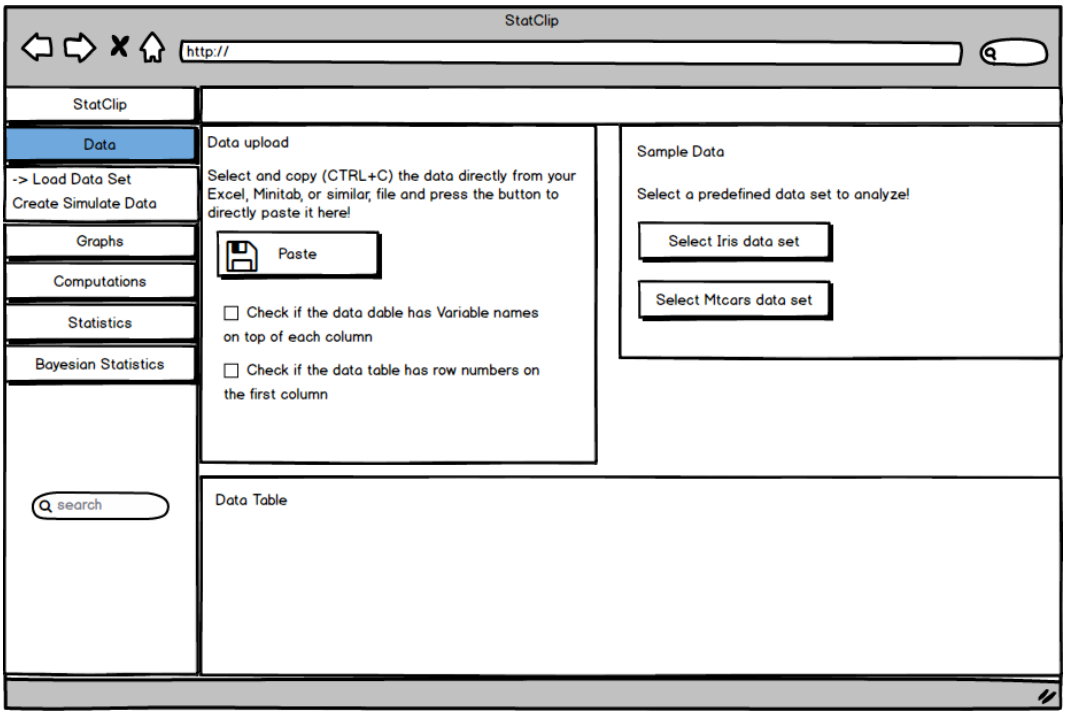


Figura 6.4 Aparença de la secció *Data*. Apartat *Load Data Set*. Versió antiga amb el botó *Paste*

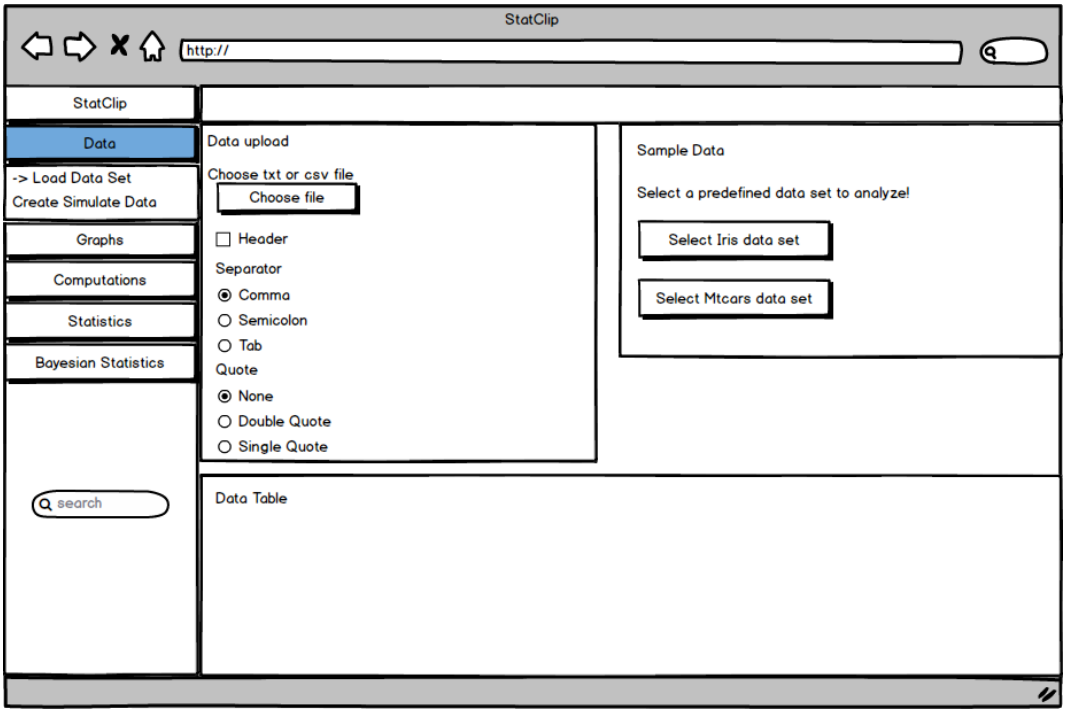


Figura 6.5 Aparença de la secció Data. Apartat Load Data Set. Versió nova amb el botó *Choose file*

6.2 Inferència

La idea inicial de l'apartat d'inferència és que l'usuari pugui escollir el model estadístic i seguidament pugui determinar la distribució a priori que serà representada. També es volen representar les distribucions predictives i un quadre resum de les estimacions puntuals. Un cop l'usuari hagi determinat les distribucions a priori i hagi introduït les dades, mitjançant el botó *update* apareixeran els gràfics de les distribucions a posteriori conjuntament amb la seva taula resum. Encara que l'usuari realitzi canvis en els *inputs* de la distribució a priori la distribució a posteriori no es veurà modificada fins que no es cliqui el botó *update*.

6.2.1 Aparença

Inicialment, per fer inferència Bayesiana és necessari escollir el model estadístic i la distribució a priori. Així doncs, es decideix que a partir d'un model estadístic escollit, l'usuari haurà de decidir els paràmetres de la distribució a priori.

Per tal d'escollir els paràmetres de la distribució a priori inicialment es va pensar en escollir directament els paràmetres, però com que en la majoria de casos els paràmetres per si sols són de difícil interpretació es decideix que l'usuari escollirà l'esperança i la desviació estàndard, conceptes més entenedors globalment. Es pot observar la presa de decisió en la Figura 6.6.

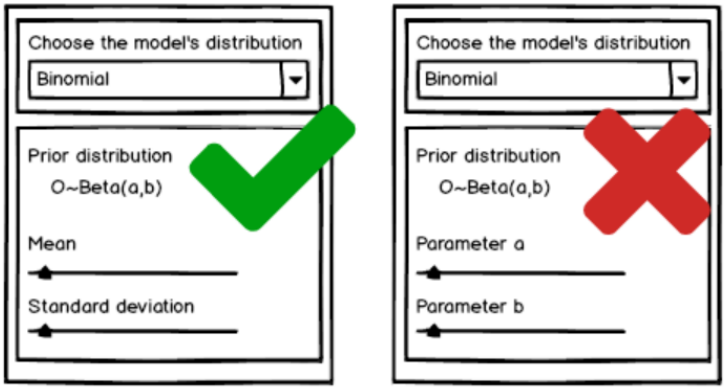


Figura 6.6 Especificació general de les distribucions a priori

Els models més representatius en la docència són el $Binomial$, el $Poisson$ i el $Normal$. Per aquest motiu aquest són els models que s'implementaran en l'aplicació *BayesClip*. Ja que els *inputs* necessaris per especificar els models no són exactament iguals es van decidir les organitzacions per l'apartat de *Configuration Options* que s'observen en la Figura 6.7.

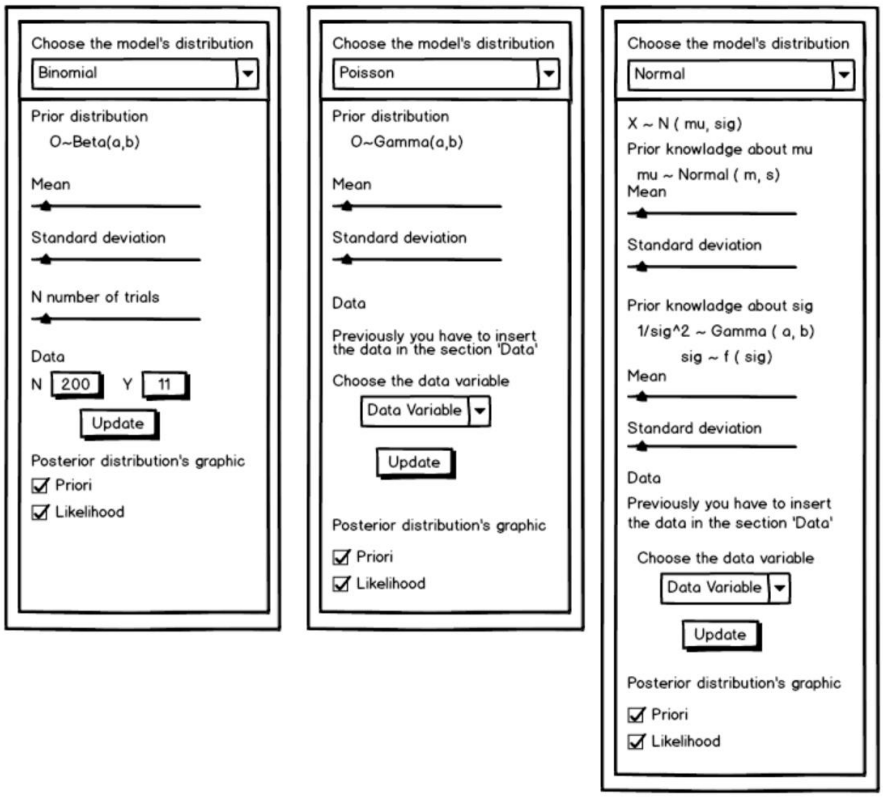


Figura 6.7 Inputs en la part de *Configuration Options* pels diferents models estadistics

Seguidament es decideix que els *outputs* de l'apartat d'inferència que es mostren a continuació seran els mateixos per tots els models estadístics:

- Un gràfic de la distribució a priori.
- Un gràfic de la distribució a posteriori. Opcionalment es pot superposar la versemblança i la distribució a priori.
- Un gràfic de la distribució predictiva a priori.
- Un gràfic de la distribució predictiva a posteriori.
- Una taula resum de les estimacions puntuals dels paràmetres, la mitjana, la variància i la mediana de les distribucions.
- Una taula amb els resultats dels intervals de credibilitat per a cada distribució.

Tal com s'ha dit anteriorment a l'inici d'aquest punt, *BayesClip* ha de seguir l'aparença de *StatClip*. Però després de decidir els outputs de la aplicació s'observa que l'espai de l'apartat *Results* no és suficient. Per tal de que la aplicació es pugui veure en una pantalla d'ordinador sense la necessitat de moure-la verticalment es decideix incorporar un *tabPanel* a l'apartat de *Results*. És a dir, s'incorporarà una caixa amb pestanyes, tal com mostra la Figura 6.8.

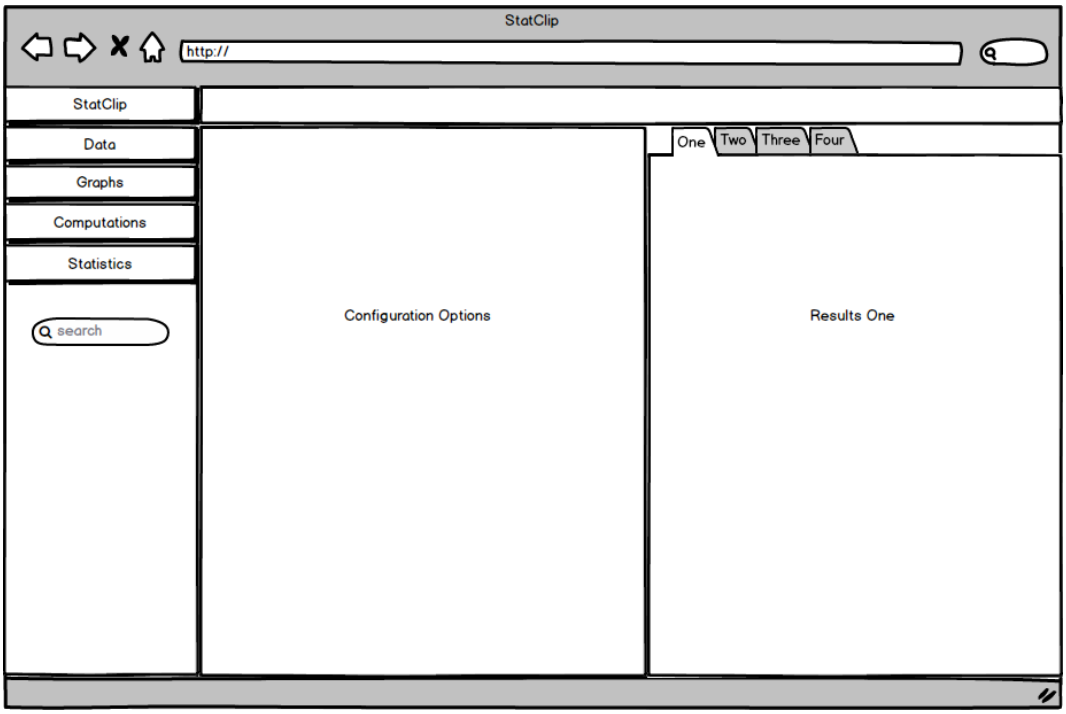


Figura 6.8 Aparença bàsica de *BayesClip*. Part de *Results* incorporada amb pestanyes

Les pestanyes de la part de *Results* i els *outputs* que inclouen són:

- *Estimation*: conté els gràfics de la distribució a priori i posteriori.
- *Predictive*: conté els gràfics de les distribucions predictives.
- *Summary*: conté la taula resum i els intervals de confiança de les distribucions.
- *Convergence*: conté la convergència de l'algoritme en el cas de les distribucions no conjugades.

El disseny resultant de les pestanyes s'observa en la Figura 6.9, la Figura 6.10, la Figura 6.11 i la Figura 6.12.

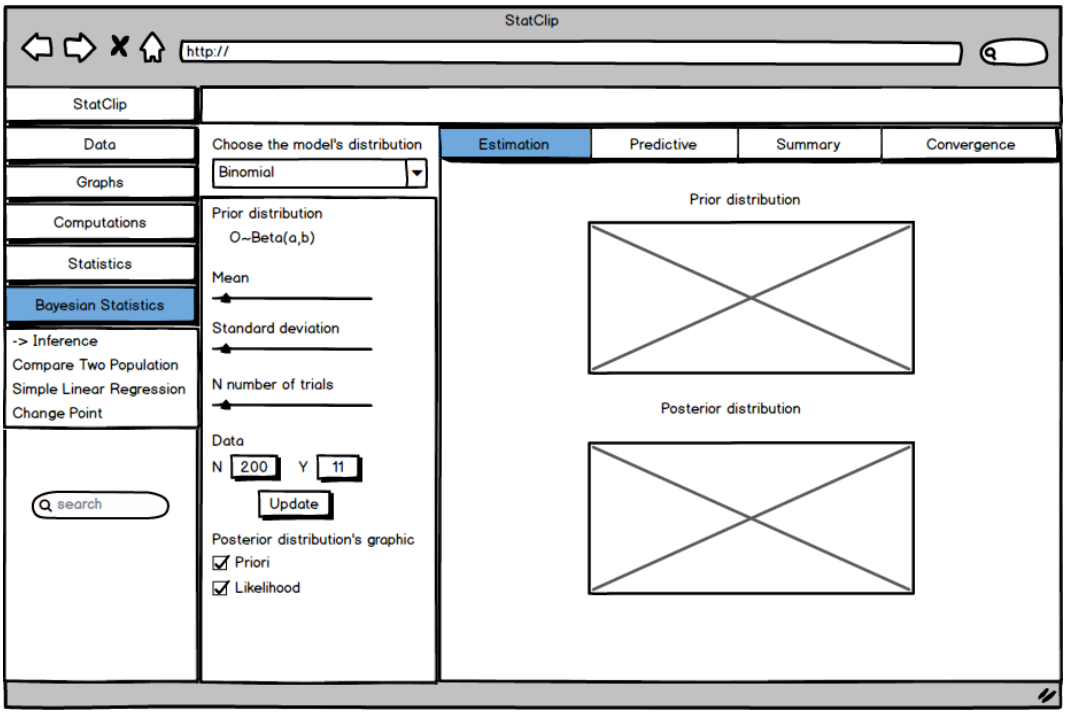


Figura 6.9 Aparença de l'apartat *Inference* en la pestanya *Estimacion* amb el model *Binomial*

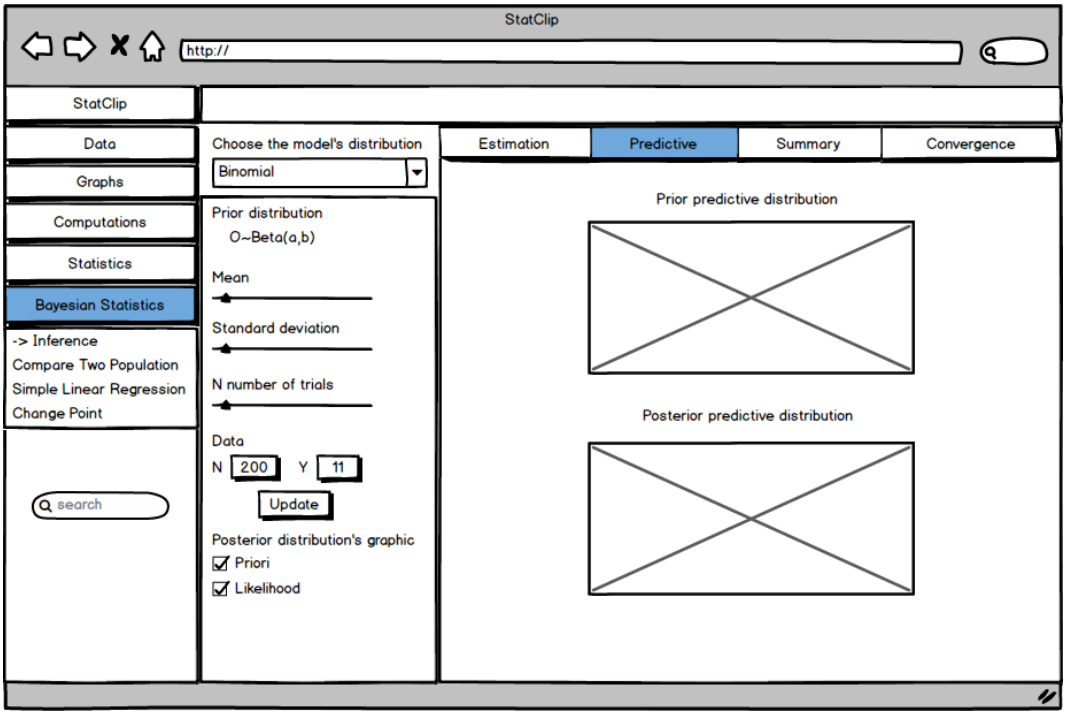


Figura 6.10 Aparença de l'apartat *Inference* en la pestanya *Predictive* amb el model *Binomial*

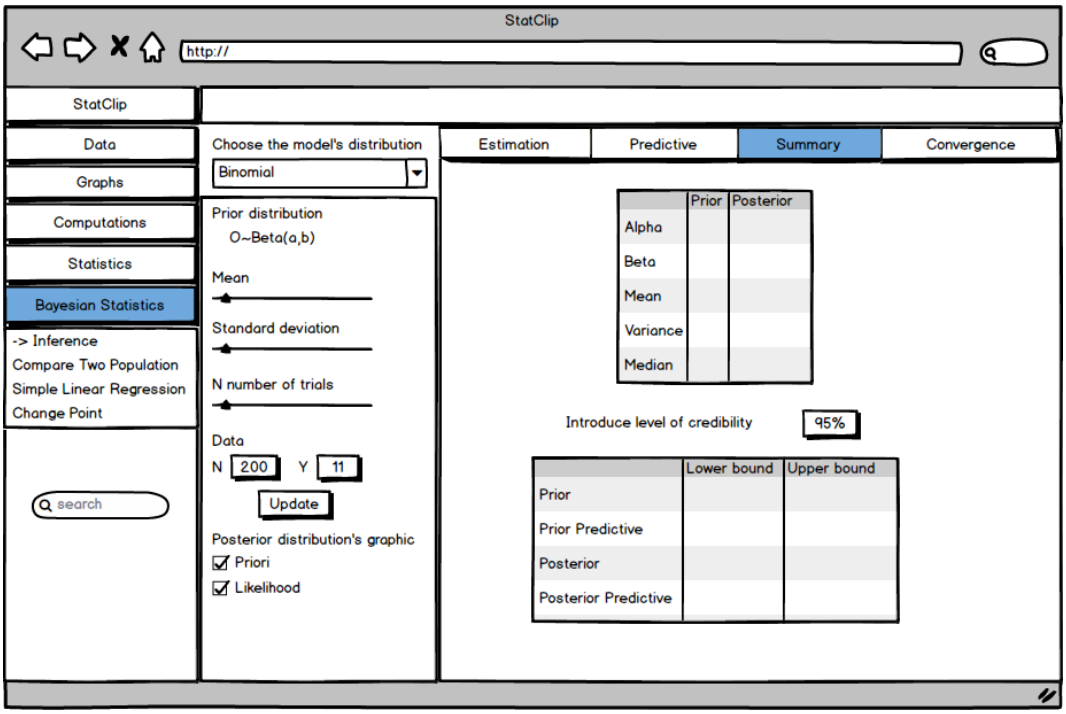


Figura 6.11 Aparença de l'apartat *Inference* en la pestanya *Summary* amb el model *Binomial*

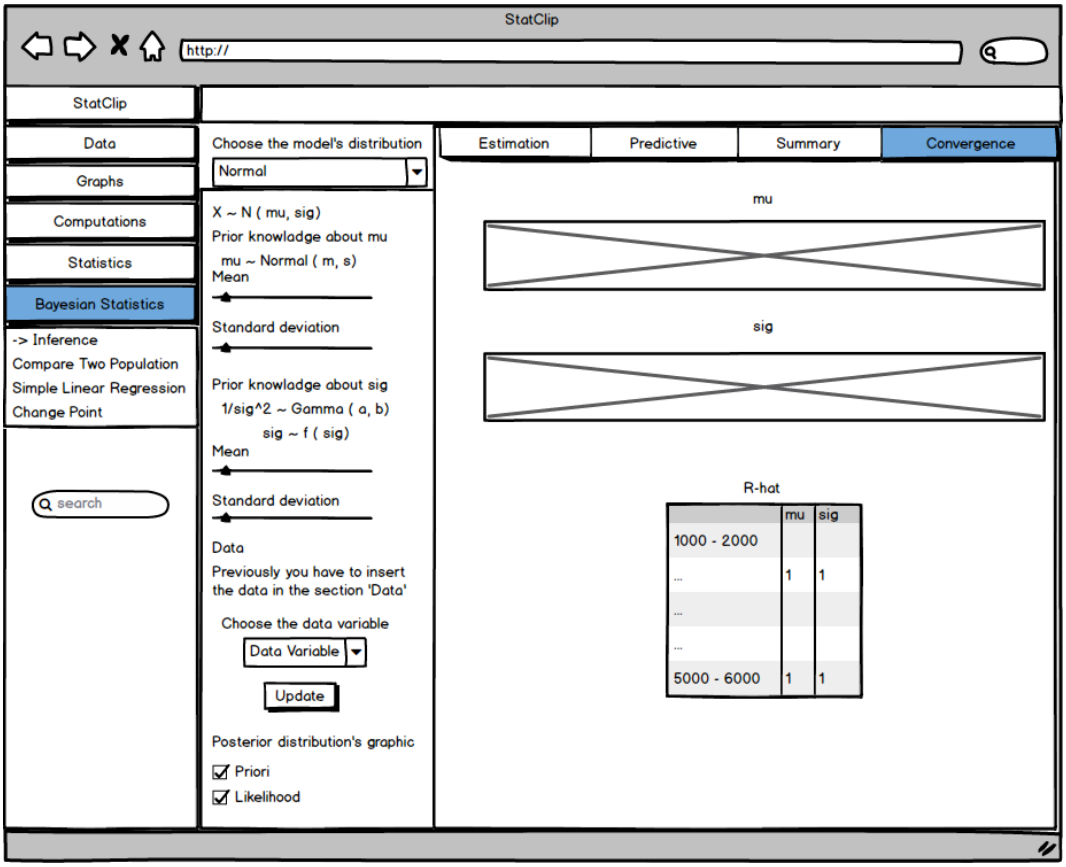


Figura 6.12 Aparença de l'apartat *Inference* en la pestanya *Convergence* amb el model *Normal*

6.2.2 Implementació

Tal com s'ha indicat anteriorment, segons el model estadístic escollit per l'usuari s'han de mostrar diferents *inputs*. Per implementar els *inputs* serà necessari fer-ho en un fitxer *server.r* degut a l'estructura interna de *StatClip*.

Per tal d'elaborar una implementació més clara i entenedora, s'elaborarà cada model estadístic en un arxiu *server.r* diferent, de tal forma que hi haurà un arxiu global per l'apartat d'inferència que cridarà la resta d'arxius com es pot observar en la Figura 6.13.

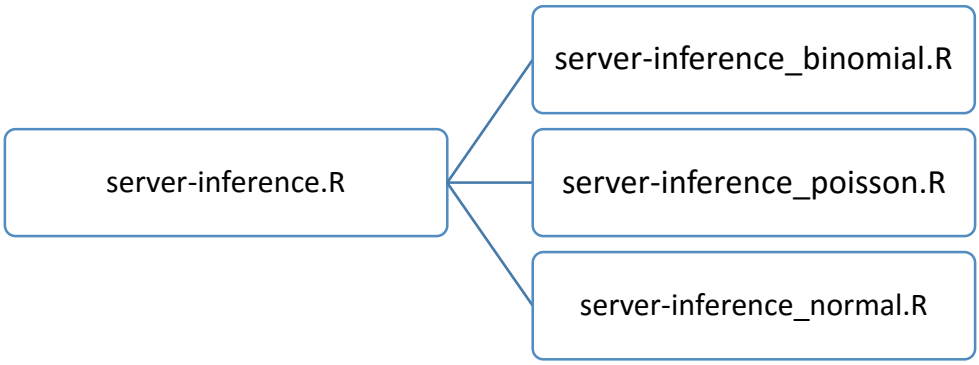


Figura 6.13 Estructura dels arxius per la implementació de l'apartat *Inference*

- **Model Binomial**

Per tal d'estimar el paràmetre θ del model estadístic $Binomial(\theta, m)$ és necessària una distribució a priori $Beta(a, b)$, ja que el paràmetre θ està definit en (0,1).

Per poder dur a terme els gràfics de les distribucions cal indicar a l'*R* els paràmetres del model, per obtenir-los a partir de l'esperança i desviació típica introduïda per l'usuari cal resoldre un sistema d'equacions. Així si $\theta \sim Beta(a, b)$ aleshores es té:

$$E(\theta) = \frac{a}{a+b}, \qquad V(\theta) = \frac{ab}{(a+b)^2(a+b+1)}.$$

Per relaxar la notació es denota $\mathbb{E} \coloneqq E(\theta)$ i $\mathbb{V} \coloneqq V(\theta)$.

A partir de les equacions de l'esperança i la variància en funció dels paràmetres es procedeix a aïllar els paràmetres de la següent forma:

$$\mathbb{E} = \frac{a}{a+b} \Rightarrow \mathbb{E}(a+b) = a \Rightarrow \begin{cases} a+b = \dfrac{a}{\mathbb{E}}; \\ \mathbb{E}a + \mathbb{E}b = a \Rightarrow \mathbb{E}b = a - \mathbb{E}a \Rightarrow b = \dfrac{a-\mathbb{E}a}{\mathbb{E}} \Rightarrow b = \dfrac{a(1-\mathbb{E})}{\mathbb{E}}; \end{cases}$$

Un cop aïllat el paràmetre b de l'equació de l'esperança, mitjançant l'equació de la variància es procedeix a aïllar el paràmetre a:

$$\mathbb{V} = \frac{ab}{(a+b)^2(a+b+1)} \Rightarrow \mathbb{V} = \frac{a}{(a+b)}\frac{b}{(a+b)(a+b+1)},$$

aleshores, utilitzant la igualtat $a + b = \frac{a}{\mathbb{E}}$ s'obté:

$$\mathbb{V} = \mathbb{E}\frac{b}{\frac{a}{\mathbb{E}}\left(\frac{a}{\mathbb{E}}+1\right)} \Rightarrow \mathbb{V} = \mathbb{E}\frac{\left(\frac{a(1-\mathbb{E})}{\mathbb{E}}\right)}{\frac{a}{\mathbb{E}}\left(\frac{a}{\mathbb{E}}+1\right)} \Rightarrow \mathbb{V} = \frac{\mathbb{E}(1-\mathbb{E})}{\left(\frac{a}{\mathbb{E}}+1\right)} \Rightarrow \mathbb{V}\left(\frac{a}{\mathbb{E}}+1\right) = \mathbb{E}(1-\mathbb{E}) \Rightarrow \frac{a}{\mathbb{E}}+1$$
$$= \frac{\mathbb{E}(1-\mathbb{E})}{\mathbb{V}} \Rightarrow a = \left(\frac{\mathbb{E}(1-\mathbb{E})}{\mathbb{V}}-1\right)\mathbb{E} \Rightarrow a = \frac{\mathbb{E}^2(1-\mathbb{E})}{\mathbb{V}} - \mathbb{E}.$$

Així doncs a partir de l'esperança i la desviació tipus introduïda per l'usuari i les equacions $b = \frac{a(1-\mathbb{E})}{\mathbb{E}}$ i $a = \frac{\mathbb{E}^2(1-\mathbb{E})}{\mathbb{V}} - \mathbb{E}$ s'obtenen els paràmetres a i b per introduir a la funció del *R*.

Els límits d'elecció de l'esperança són entre 0 i 1, ja que la distribució $Beta$ es mou en aquest interval. Per definir els límits de la variància s'ha d'acotar l'equació anterior de la següent forma:

$$a = \frac{\mathbb{E}^2(1-\mathbb{E})}{\mathbb{V}} - \mathbb{E} \Rightarrow a = \frac{\mathbb{E}^2 - \mathbb{E}^3 - \mathbb{V}\mathbb{E}}{\mathbb{V}} > 0 \; ja \; que \; a \in (0,\infty) \; i \; \mathbb{V} > 0, \Rightarrow \mathbb{E}^2 - \mathbb{E}^3 - \mathbb{V}\mathbb{E} > 0$$
$$\Rightarrow \mathbb{E}^2(1-\mathbb{E}) - \mathbb{V}\mathbb{E} > 0 \Rightarrow \mathbb{E}(\mathbb{E}(1-\mathbb{E}) - \mathbb{V}) > 0 \Rightarrow \mathbb{E}(1-\mathbb{E}) - \mathbb{V} > 0 \Rightarrow \mathbb{V}$$
$$< \mathbb{E}(1-\mathbb{E}) \; on \; \mathbb{E} \in (0,1).$$

Així doncs, a partir d'una esperança donada entre 0 i 1, la variància es trobarà entre 0 i $\mathbb{E}(1-\mathbb{E})$.

El codi que implementa els elements del model Binomial que s'observen en la Figura 6.7 es troba a l'Annex.Seguidament es procedeix a implementar els elements de l'apartat *Results* organitzats en pestanyes.

En la primera pestanya es troba el gràfic de la distribució a priori. Com que els paràmetres de la distribució s'obtenen com a solució de dues equacions que depenen de l'esperança i de la desviació típica, la resolució d'aquestes introdueix un error numèric de precisió, de manera que en els casos particulars en què la Beta és una Uniforme, l'*R* dibuixa una distribució quasi uniforme, però no exactament uniforme. Així doncs per solucionar-ho s'incorpora una avaluació del paràmetre, si aquest és molt pròxim a 1 s'aproxima directament a 1.

En la primera pestanya també es troba el gràfic de la distribució a posteriori. Opcionalment l'usuari podrà incorporar la distribució a priori i la versemblança en aquest mateix gràfic mitjançant un *checkboxGroup,* que és un quadre amb diverses opcions per seleccionar. Així

doncs la funció per representar el gràfic, *plot()*,incorporarà les línies de les distribucions seleccionades en el quadre.

En la pestanya *Predictive* es troben els gràfics de la distribució predictiva a priori i a posteriori. Al ser un model estadístic conjugat les distribucions predictives són exactes i no es necessària la simulació. La distribució conjugada de la $Beta$ és la $BetaBinomial$. Així doncs, s'utilitza la funció *dbetabinom.ab* per representar la distribució predictiva. Aquesta funció es troba dins del paquet *VGAM* i permet definir la distribució amb els paràmetres a i b.

En la pestanya *Estimation* es crea una taula amb les estimacions puntuals i per interval decidides anteriorment. Finalment, en aquest cas la pestanya *Convergence* està buida ja que es tracta d'un model conjugat.

- **Model Poisson**

Per tal d'estimar el paràmetre θ del model estadístic $Poisson(\theta)$ és necessària una distribució a priori $Gamma(a,b)$, ja que el paràmetre θ està definit en els positius.

Per poder dur a terme els gràfics de les distribucions cal indicar a l'*R* els paràmetres, per obtenir-los a partir de l'esperança i desviació típica introduïda per l'usuari cal resoldre un sistema d'equacions. Així si $\theta \sim Gamma(a,b)$, llavors es té que:

$$E(\theta) = \frac{a}{b}, \qquad V(\theta) = \frac{a}{b^2}.$$

Seguidament un petit canvi de nomenclatura per abreviar: $\mathbb{E} \coloneqq E(\theta)$ i $\mathbb{V} \coloneqq V(\theta)$.

A partir de les equacions de l'esperança i la variància s'aïllen els paràmetres.

$$\mathbb{E} = \frac{a}{b} \Rightarrow b\mathbb{E} = a\,.$$
$$\mathbb{V} = \frac{a}{b^2} \Rightarrow \mathbb{V} = \frac{b\mathbb{E}}{b^2} \Rightarrow \mathbb{V} = \frac{\mathbb{E}}{b} \Rightarrow b = \frac{\mathbb{E}}{\mathbb{V}}.$$
$$a = \mathbb{E}\frac{\mathbb{E}}{\mathbb{V}} \Rightarrow a = \frac{\mathbb{E}^2}{\mathbb{V}}.$$

Així doncs a partir de l'esperança i la desviació tipus introduïda i les equacions $a = \frac{\mathbb{E}^2}{\mathbb{V}}$ i $b = \frac{\mathbb{E}}{\mathbb{V}}$ s'obtenen els paràmetres a i b per introduir a la funció del *R*.

Els intervals d'elecció de l'esperança i la desviació tipus es posen en (0,20) i (0,10) respectivament, ja que l'usuari resoldrà exercicis preparats i no es necessari un interval de valors més gran.

Per introduir les dades del model es decideix col·locar un desplegable per tal de seleccionar la variable on es troben les dades. Prèviament l'usuari haurà d'haver introduït la base de dades en l'apartat *Data*.

Seguidament s'utilitza una funció *reactive* per tal d'obtenir els paràmetres de la funció Gamma, els quals es tornaran a calcular només quan es canviïn els valors de l'esperança i la desviació tipus de la distribució a priori. D'aquesta forma s'intentarà optimitzar el temps de computació.

Per tal de representar el gràfic de la distribució a priori i el gràfic de la distribució a posteriori (on opcionalment es podrà representar la versemblança i la priori) s'utilitzarà el mateix codi que en el model $Binomial$ però canviant les distribucions per les del model $Poisson$.

En aquest cas, en la pestanya *Predictive* s'utilitza la distribució predictiva $BinomialNegativa$.

La implementació del codi de la part *Predictive* i *Summary* és exactament igual que en el model $Binomial$ però canviant les distribucions per les del model $Poisson$.

- ## Model Normal

En el model $Normal\ (\mu,\ \sigma)$ són necessàries dues distribucions a priori: una pel paràmetre μ i l'altra pel paràmetre σ.

Normalment, en estadística Bayesiana s'utilitza una distribució $Normal$ per estimar μ i una distribució $Gamma$ per estimar la precisió, $\tau = \frac{1}{\sigma^2}$. En aquest cas es decideix estimar la desviació tipus enlloc de la precisió, ja que com a concepte és més entenedor, o si més no és al que estan acostumats la majoria d'usuaris.

Per estimar σ és necessari trobar la distribució a priori aplicant un canvi de variable de la següent forma. Si $\tau \sim Gamma(a,b)$ amb funció de densitat $f(\tau) = \frac{b^a}{\Gamma(a)}\tau^{a-1}e^{-b\tau}I_{[0,\infty)}(\tau)$, aleshores es pot obtenir la funció de densitat de σ a través del canvi de variable:

$$\sigma = g(\tau) = \frac{1}{\sqrt{\tau}} \quad \text{i} \quad g^{-1}(\sigma) = \frac{1}{\sigma^2}.$$

Finalment, la funció de densitat de σ és:

$$f(\sigma) = f\left(g^{-1}(\sigma)\right)\left|J\left(g^{-1}(\sigma)\right)\right| = f\left(g^{-1}(\sigma)\right)\left|\frac{-2}{\sigma^3}\right| = \frac{2}{\sigma^3}\frac{b^a}{\Gamma(a)}\left(\frac{1}{\sigma^2}\right)^{a-1} e^{\frac{-b}{\sigma^2}},\ \text{amb}\ \sigma > 0.$$

Un cop trobada la distribució que segueix la desviació tipus, és necessari trobar la seva esperança i variància. Per trobar l'esperança de σ es resol la següent integral.

$$E(\sigma) = \int_0^\infty y\, f_\sigma(y)dy = \int_0^\infty \frac{2y}{y^3}\frac{b^a}{\Gamma(a)}\left(\frac{1}{y^2}\right)^{a-1} e^{\frac{-b}{y^2}}\, dy.$$

S'aplica el canvi de variable $t = \frac{b}{y^2}$, $y = \sqrt{\frac{b}{t}}$, $dt = \frac{-2b}{y^3}dy$ i s'obté:

$$E(\sigma) = \frac{1}{-b}\int_0^{\infty} \frac{2y}{y^3}\frac{b^a}{\Gamma(a)}\left(\frac{1}{y^2}\right)^{a-1} e^{\frac{-b}{y^2}}(-b)dy$$

$$= \frac{1}{-b}\int_{\infty}^{0} \left(\frac{b}{t}\right)^{\frac{1}{2}}\frac{b^a}{\Gamma(a)}\left(\frac{t}{b}\right)^{a-1} e^{-t}dt = \frac{1}{b}\frac{b^a}{\Gamma(a)}\frac{b^{\frac{1}{2}}}{b^{a-1}}\int_0^{\infty}\left(\frac{1}{t}\right)^{\frac{1}{2}} t^{a-1}e^{-t}dt$$

$$= \frac{b^{\frac{1}{2}}}{\Gamma(a)}\int_0^{\infty} t^{a-\frac{3}{2}}e^{-t}dt = \frac{b^{\frac{1}{2}}}{\Gamma(a)}\Gamma\left(a-\frac{1}{2}\right).$$

Un cop trobada l'esperança es vol calcular la variància. Per calcular la variància s'utilitza la fórmula $V(\sigma) = E(\sigma^2) - E(\sigma)^2$.

Així doncs, és necessari calcular l'esperança de σ^2 amb la integral

$$E(\sigma^2) = \int_0^{\infty} y^2 f_\sigma(y)dy = \int_0^{\infty} \frac{2y^2}{y^3}\frac{b^a}{\Gamma(a)}\left(\frac{1}{y^2}\right)^{a-1} e^{\frac{-b}{y^2}}\, dy.$$

S'aplica el canvi de variable $t = \frac{b}{y^2},\ y = \sqrt{\frac{b}{t}},\ \ dt = \frac{-2b}{y^3}dy$ i s'obté:

$$E(\sigma^2) = \frac{1}{-b}\int_0^{\infty} \frac{2y^2}{y^3}\frac{b^a}{\Gamma(a)}\left(\frac{1}{y^2}\right)^{a-1} e^{\frac{-b}{y^2}}(-b)dy$$

$$= \frac{1}{-b}\frac{b^a}{\Gamma(a)}\int_{\infty}^{0}\frac{b}{t}\left(\frac{t}{b}\right)^{a-1} e^{-t}dt = \frac{b}{\Gamma(a)}\int_0^{\infty} t^{a-2}\, e^{-t}dt = \frac{b}{\Gamma(a)}\Gamma(a-1)$$

$$= \frac{b\Gamma(a-1)}{\Gamma(a-1)(a-1)} = \frac{b}{a-1}.$$

Així doncs la variància de σ és:

$$V(\sigma) = E(\sigma^2) - E(\sigma)^2 = \frac{b}{a-1} - \left(\frac{b^{\frac{1}{2}}}{\Gamma(a)}\Gamma\left(a-\frac{1}{2}\right)\right)^2 = \frac{b}{a-1} - \frac{b\Gamma\left(a-\frac{1}{2}\right)^2}{\Gamma(a)^2}.$$

Per relaxar la notació es denota $\mathbb{E} \coloneqq E(\theta)$ i $\mathbb{V} \coloneqq V(\theta)$**.** Un cop trobades les equacions de l'esperança i la variància

$$\begin{cases} \mathbb{V} = \dfrac{b}{a-1} - \dfrac{b\Gamma\left(a-\frac{1}{2}\right)^2}{\Gamma(a)^2}, \\ \mathbb{E} = \dfrac{b^{\frac{1}{2}}}{\Gamma(a)}\Gamma\left(a-\frac{1}{2}\right), \end{cases}$$

es substitueix la segona equació a la primera i s'obté el paràmetre b:

$$\mathbb{V} = \frac{b}{a-1} - \frac{b\Gamma\left(a-\frac{1}{2}\right)^2}{\Gamma(a)^2} \Rightarrow \mathbb{V} = \frac{b}{a-1} - \mathbb{E}^2 \Rightarrow \mathbb{V} + \mathbb{E}^2 = \frac{b}{a-1} \Rightarrow b = (a-1)(\mathbb{V} + \mathbb{E}^2).$$

Seguidament es substitueix aquesta expressió de b a l'equació inicial de l'esperança d'aquesta forma:

$$\mathbb{E} = \frac{b^{\frac{1}{2}}}{\Gamma(a)}\Gamma\left(a-\frac{1}{2}\right) \Rightarrow b^{\frac{1}{2}} = \frac{\mathbb{E}\Gamma(a)}{\Gamma\left(a-\frac{1}{2}\right)} \Rightarrow\ b = \frac{\mathbb{E}^2\Gamma(a)^2}{\Gamma\left(a-\frac{1}{2}\right)^2} \Rightarrow (a-1)(\mathbb{V}+\mathbb{E}^2) = \frac{\mathbb{E}^2\Gamma(a)^2}{\Gamma\left(a-\frac{1}{2}\right)^2}$$
$$\Rightarrow \frac{\mathbb{E}^2\Gamma(a)^2}{\Gamma\left(a-\frac{1}{2}\right)^2} - (a-1)(\mathbb{V}+\mathbb{E}^2) = 0.$$

En aquesta expressió no es pot aïllar a de forma analítica i per calcular-la s'ha de recórrer a mètodes numèrics. Mitjançant el mètode de la bisecció es trobarà el paràmetre a per després calcular el paràmetre b.

Un cop calculades l'esperança i la variància de σ és necessari decidir quin és l'interval de valors en el que es deixa triar l'usuari. Per la complexitat de resolució del sistema d'equacions i la necessitat d'aplicar mètodes numèrics es decideixen els intervals $E(\sigma) \in [0.75, 6]$ i $\sqrt{V(\sigma)} \in [0.31, 0.5E(\sigma)]$ pel correcte funcionament de l'aplicació.

Es decideix que l'esperança i la variància de μ es mouran entre $(-10, 10)$ i $(0, 10)$ respectivament, ja que no són necessaris més valors resoldre els exercicis preparats.

Les dades numèriques s'introduiran prèviament en *Data* i es seleccionaran mitjançant un desplegable.

La representació de les distribucions a priori no suposa cap complexitat afegida, un cop calculats els paràmetres de les distribucions.

Per tal de representar la distribució a posteriori és necessari aplicar l'algoritme del *Gibbs Sampling* per la distribució $Normal$. Per implementar-lo es parteix de que les dades es defineixen com $z = \{x_1, \dots, x_n\}$ amb $-\infty < x_i < \infty$ i la seva funció de densitat $P(x_i|\mu, \tau) = N(x_i|\mu, \tau)$ amb $-\infty < \mu < \infty$ i $\tau > 0$, on τ és la precisió definida com la inversa de la variància, $\tau = \frac{1}{\sigma^2}$. Aleshores la distribució a priori pel paràmetre μ és $\pi(\mu) = N(\mu|m_0, s_0)$ i la distribució a posteriori és $\pi(\mu|z) = N(\mu|m_n, s_n)$ on $m_n = s_n^{-1}(s_0 m_0 + n\tau\bar{x})$, $s_n = s_0 + n\tau$ i $\bar{x} = n^{-1}\sum_{i=1}^{n} x_i$. La distribució a priori pel paràmetre τ és $\pi(\tau) = Gamma(\tau|a, \mathrm{b})$ i la distribució a posteriori és $\pi(\tau|z) = Gamma\left(\tau \middle| a + \frac{1}{2}n, b + \frac{1}{2}t\right)$ on $t = \sum_{i=1}^{n}(x_i - \mu)^2$.

A partir de les dades, dels paràmetres de la distribució a priori per μ inicials, m_0 i s_0, i els paràmetres de la distribució a priori per τ, a i b, es procedeix a iniciar l'algoritme. Primerament es determinen uns valors arbitraris per $\mu^{(0)}$ i $\tau^{(0)}$ i es calcula un valor de cada distribució a posteriori, $\mu^{(1)}$ i $\tau^{(1)}$. Seguidament, s'actualitzen els valors dels paràmetres de les distribucions i s'obtenen els següents valors de la cadena $\mu^{(2)}$ i $\tau^{(2)}$, i així successivament fins a la convergència de les cadenes. És a dir, fins que μ i τ estacionin en un valor. Finalment s'aplica la transformació $\sigma = \frac{1}{\sqrt{\tau}}$ a les simulacions de τ per tal d'obtenir valors de la variable desviació tipus.

Ja que aquest algoritme serà necessari en diversos càlculs, s'introdueix en una funció *reactive,* la qual només s'executa quan canvien els valors del seu interior. Com que per la part de *Convergence* serà necessari comparar dues cadenes, s'implementen dues funcions *reactive,* iniciant l'algoritme en valors inicials molt diferents.

Seguidament es pot determinar la distribució a posteriori conjuntament amb un missatge d'avís que indica que el càlcul tarda uns segons.

Per la generació de la distribució predictiva a priori, inicialment s'han de generar valors aleatoris de les distribucions a priori, $\mu \sim Normal(m_0, s_0)$ i $\sigma \sim f(\sigma)$. La generació dels valors aleatoris de μ és directa. En canvi, per generar valors aleatoris de σ s'han de generar de la precisió, $\tau = \frac{1}{\sigma^2}$ on $\tau \sim Gamma(a, b)$ (Els paràmetres de la funció Gamma i $f(\sigma)$ són els mateixos). Un cop obtinguts els valors aleatoris de μ i s es generen els valors aleatoris de la distribució predictiva a priori, $Normal(\mu, \frac{1}{\sqrt{\tau}})$.

Per la representació de la distribució predictiva a posteriori s'utilitzen les simulacions del *Gibbs Sampling* per generar valors de la distribució $Normal$. Un cop generats els valors, s'eliminen els extrems per tal de que el gràfic no sigui molt extens.

Tal com s'ha fet en els altres models estadístics, en la pestanya *Summary* s'extreuen les estimacions puntuals i per interval de les distribucions i es representen en una taula.

Finalment per la pestanya de *Convergence* es representen les dues cadenes de μ, les dues cadenes de τ i una taula amb els valors de *R-hat* per cada interval de simulacions.

6.3 Comparació de dues poblacions

La idea inicial de l'apartat de comparació de dues poblacions és que l'usuari pugui comparar les dues poblacions després d'introduir les distribucions a priori i les dades.

Només s'implementarà el model $Binomial$ ja que no es creu necessària la utilització d'altres models estadístics en aquesta part, en tot cas es pot considerar la seva implementació en un futur.

6.3.1 Aparença

L'aparença d'aquest apartat es basa en duplicar els inputs i outputs de l'apartat d'inferència amb el model $Binomial$ i afegir un apartat *Difference* on es representarà la distribució de diferència amb el seu interval de credibilitat. Visualment l'aparença de la pestanya es pot apreciar a la Figura 6.14.

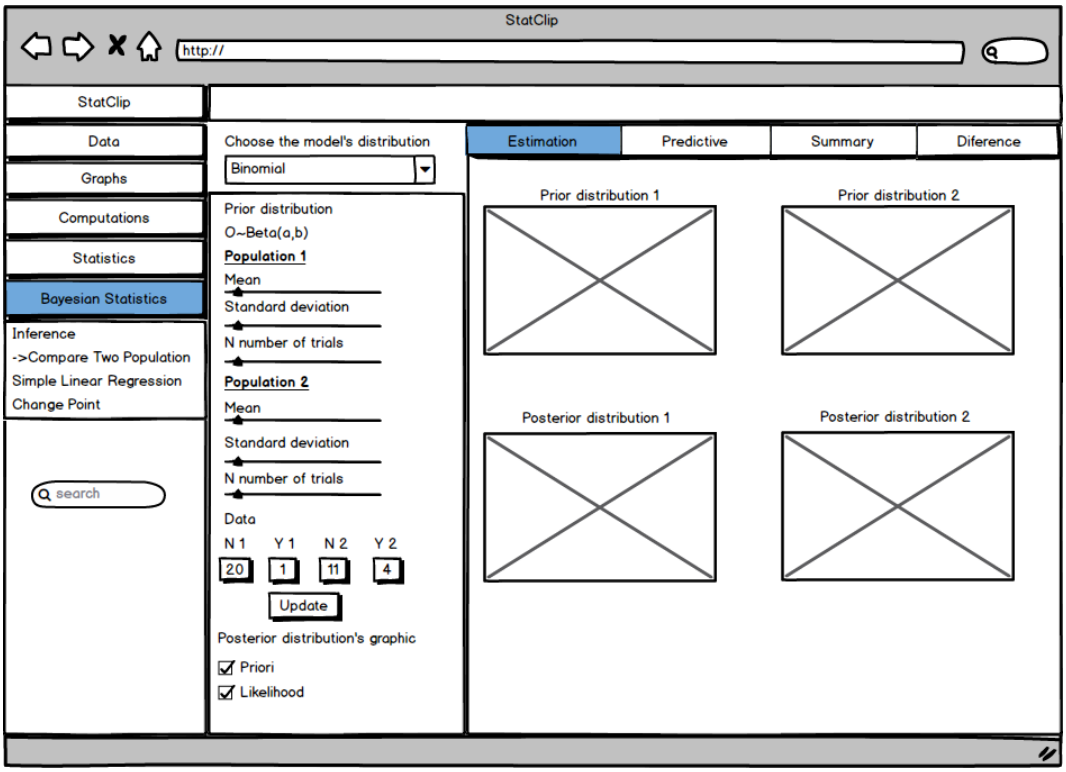


Figura 6.14 Aparença de l'apartat *Compare Two Population* en la pestanya *Estimation* amb el model *Binomial*

De la mateixa manera es troba la pestanya de *Predictive* i *Summary* en la Figura 6.15 i la Figura 6.16 respectivament.

Finalment, tal com mostra la Figura 6.17 en la pestanya de *Difference* es decideix representar la funció de densitat del logaritme del quocient de les distribucions, $\log\left(\frac{\theta_1}{\theta_2}\right)$, ja que a l'utilitzar el logaritme es suavitza més la distribució. El logaritme és igual d'entenedor que la diferència, el zero representa que les dues distribucions són iguals.

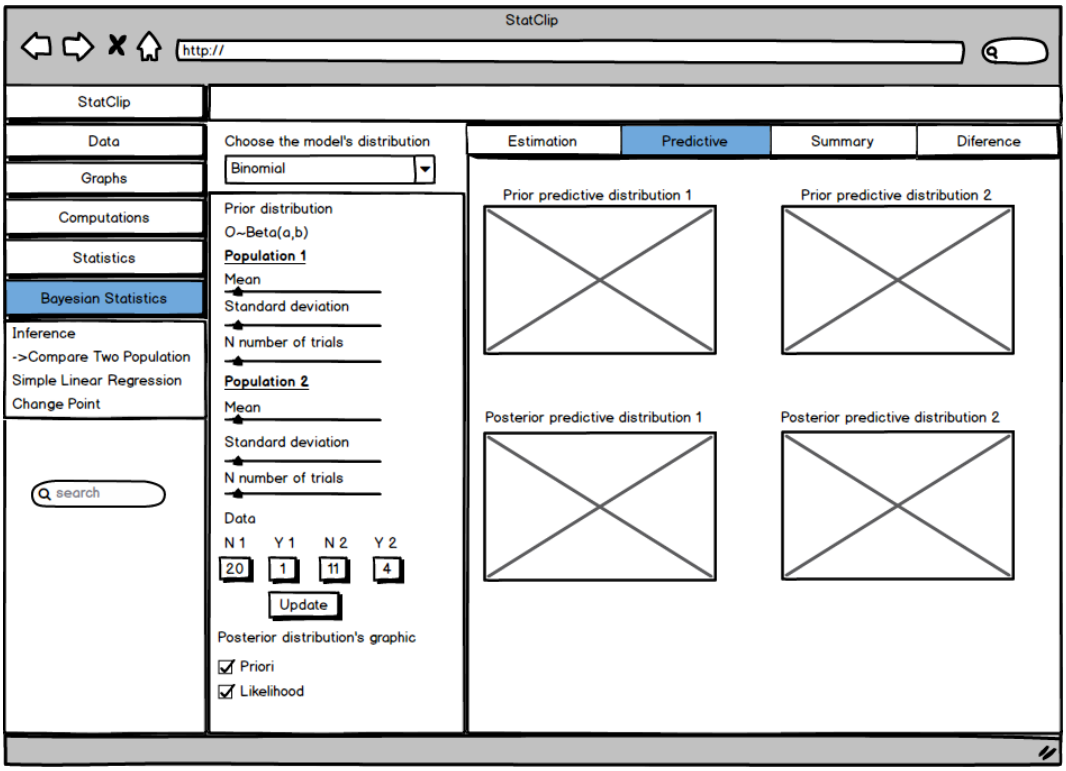


Figura 6.15 Aparença de l'apartat *Compare Two Population* en la pestanya *Predictive* amb el model *Binomial*

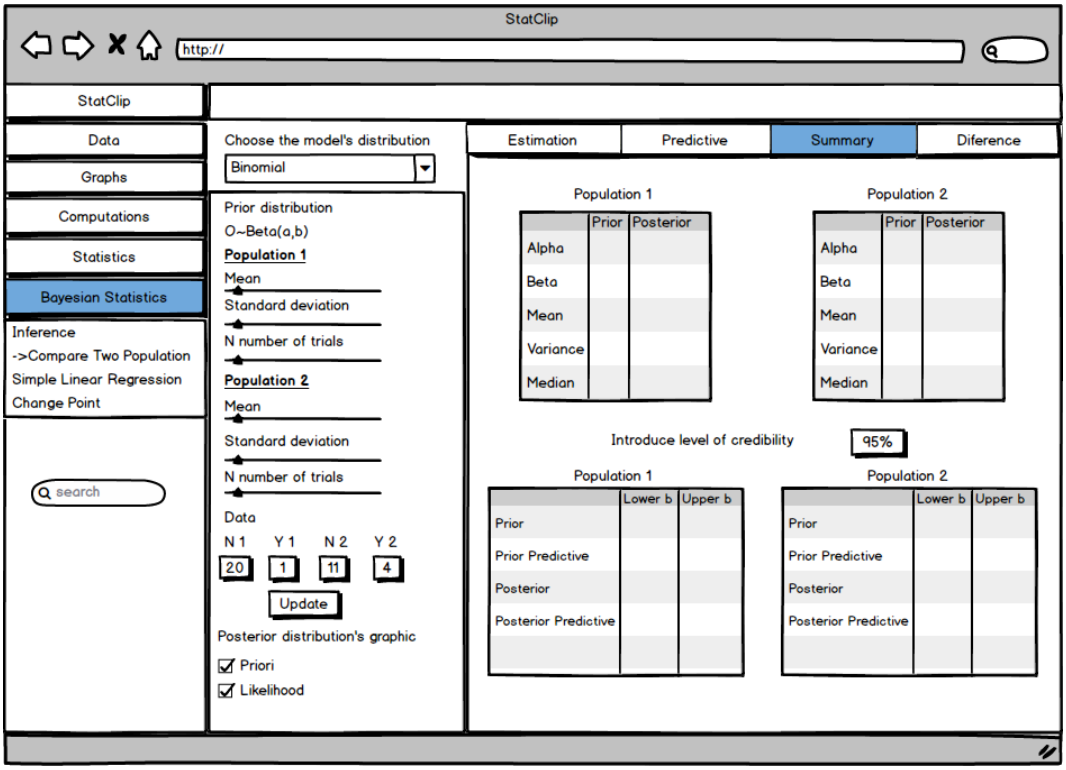


Figura 6.16 Aparença de l'apartat *Compare Two Population* en la pestanya *Summary* amb el model *Binomial*

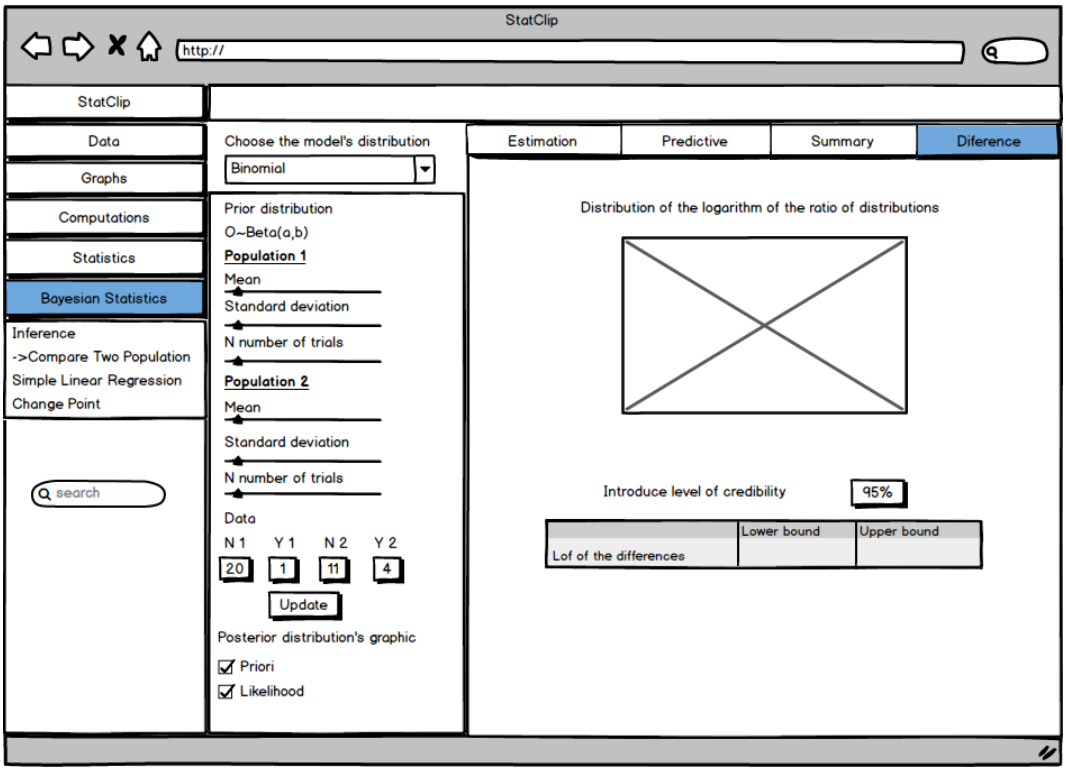


Figura 6.17 Aparença de l'apartat *Compare Two Population* en la pestanya *Diference* amb el model *Binomial*

6.3.2 Implementació

Per la implementació s'utilitza el codi de l'apartat d'inferència del model $Binomial$ duplicant els *inputs* i *outputs*.

La distribució utilitzada en la pestanya de *Difference* no és exacta, de manera que cal recórrer a la simulació. En el gràfic es decideix representar una línia vertical en el zero per tal de visualitzar millor aquest valor.

6.4 Regressió lineal simple

La idea de la regressió lineal simple és que l'usuari, a partir de les variables introduïdes, pugui estimar el model de regressió. Es contempla la possibilitat de que l'usuari pugui introduir els coneixements a priori del model. Es vol representar una taula resum amb les estimacions puntuals dels paràmetres, els gràfics de la convergència i la densitat de les distribucions. També es vol representar el núvol de punts de les dades i la recta estimada per tal de que els conceptes siguin més clars i entenedors.

Per tal d'estimar el model es decideix utilitzar una funció que es troba dins del paquet *MCMCglmm*. La funció anomenada igual que el paquet necessita les dades i, opcionalment, si es té coneixement a priori sobre els paràmetres s'ha d'introduir la matriu B (la dels efectes

fixos). La matriu B consta d'un vector amb els valors esperats i la matriu de variàncies i covariàncies dels paràmetres.

Per tal d'introduir el coneixement a priori de la variància del model s'han d'indicar els paràmetres d'una distribució $Wishart\ Inversa$. Donat la dificultat d'obtenir els paràmetres de la distribució $Wishart\ Inversa$ a partir de la esperança i la variància, i la dificultat de tenir coneixements a priori sobre la variància del model, es decideix no introduir l'opció de que l'usuari esculli a priori la variància del model, i s'utilitzarà la distribució a priori no informativa que per defecte utilitza la funció *MCMCglmm*.

6.4.1 Aparença

Es decideix que l'usuari pugui tenir coneixements o no de les distribucions a priori dels paràmetres. Així doncs mitjançant un *radioButton,* que és un quadre per escollir entre diverses opcions, es dóna la possibilitat d'introduir les distribucions a priori o no. És a dir, si l'usuari escull que sí té coneixement a priori apareixeran els requadres per introduir les dades, si l'usuari escull que no, no apareixerà res. Mitjançant dos desplegables s'escolliran les variables del model.

Igual que en l'apartat *Inference* es decideix utilitzar un panell amb pestanyes per organitzar millor els outputs. El disseny de l'apartat de regressió lineal simple amb totes les consideracions esmentades es visualitzen en la Figura 6.18 i Figura 6.19.

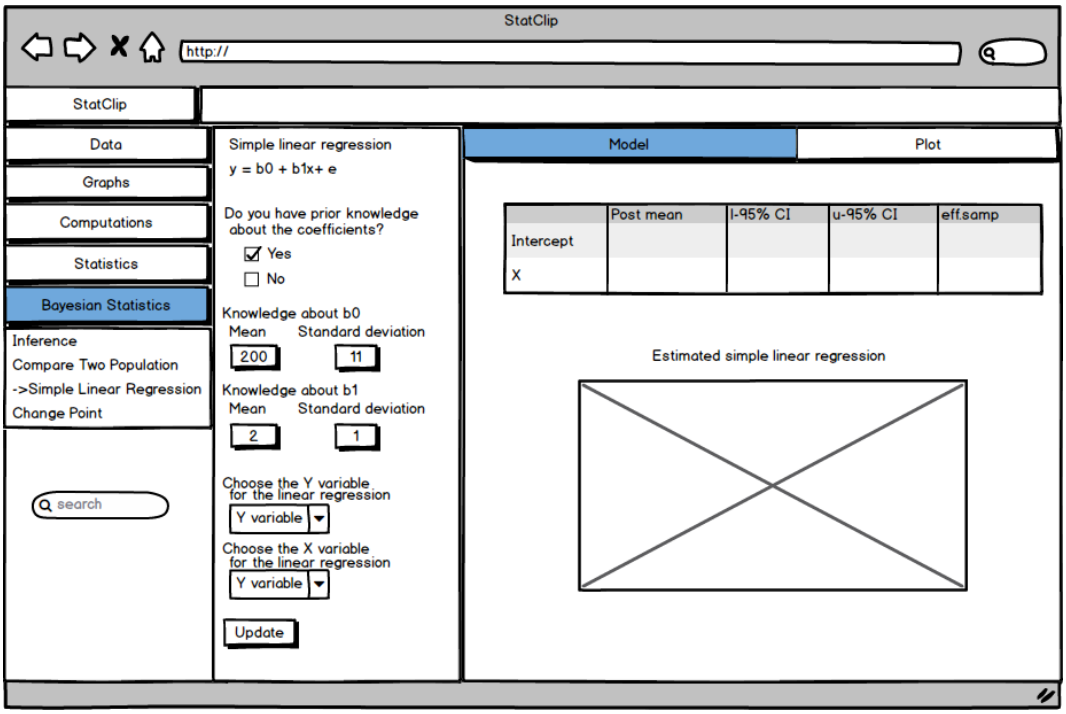


Figura 6.18 Aparença de l'apartat *Simple Linear Regression* en la pestanya *Model* amb informació a priori dels coeficients del model.

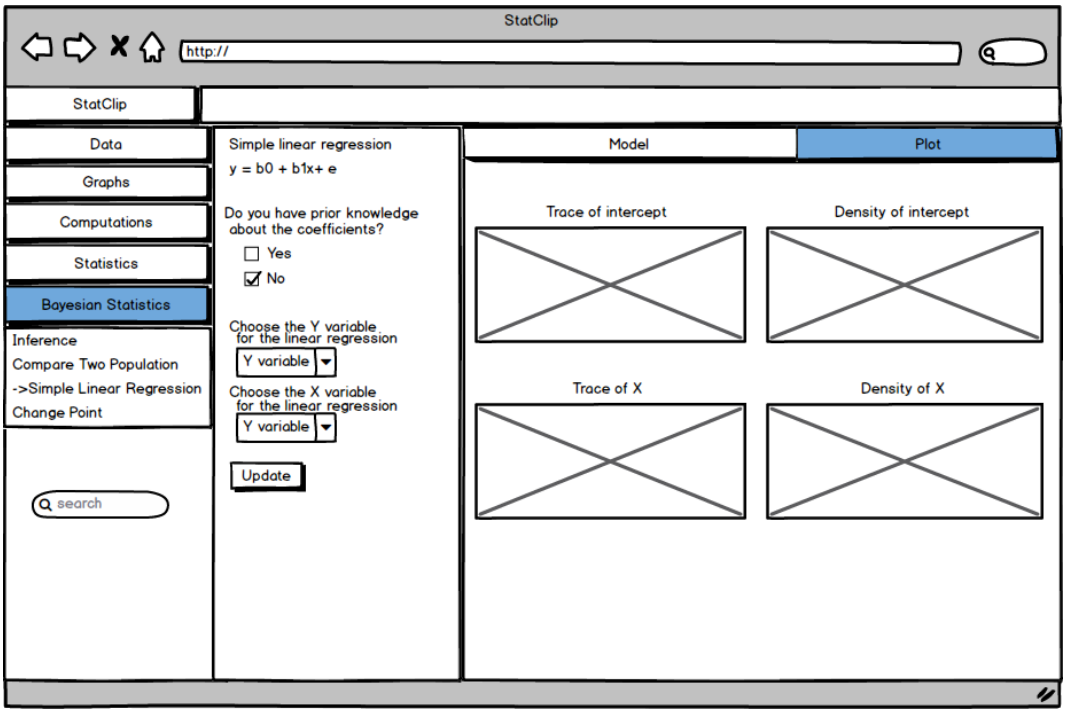


Figura 6.19 Aparença de l'apartat *Simple Linear Regression* en la pestanya *Plot* sense informació a priori dels coeficients del model.

6.4.2 Implementació

En la part de *Configuration options* en l'arxiu *UI-body_simple_linear_regression.R* s'implementen l'expressió de la regressió lineal i el *radioButton* ja que en aquest cas no són necessàries avaluacions d'objectes i no es necessari incorporar-ho en un arxiu *server.*

En l'arxiu *server-simple_linear_regression.R* s'implementen els elements necessaris per caracteritzar β_0 i β_1. En aquest cas les mitjanes i les desviacions s'indicaran amb un requadre on introduir el valor directament, enlloc d'utilitzar una barra lliscant, ja que en aquesta s'haurien de decidir el límits i depenent de les dades haurien de ser més grans o més petits.

La selecció de la variable Y i X del model s'implementa en l'arxiu *server* mitjançant dos desplegables.

Per tal d'escriure la taula resum, la funció *MCMCglmm* que estima el model ens la proporciona directament fent un *summary* de l'objecte resultant de l'estimació.

Afegint la comanda `plot(model$Sol)`, on *model* es l'objecte resultant de l'estimació, és a dir, el model, s'obtenen els plots de les funcions de densitat dels paràmetres junt amb les seves convergències.

Per tal de representar el núvol de punts i la recta estimada s'utilitza una funció per determinar la recta segons els paràmetres i seguidament es determinen els límits del gràfic segons les

dades per tal d'observar sempre el núvol de punts en el gràfic, ja que a vegades depenent de la distribució a priori escollida la recta estimada es trobarà lluny del núvol de punts.

6.5 Punt de canvi

A partir d'un model estadístic $M = \{P(y|\theta); \theta \in \Omega\}$ amb $\theta = (\theta_1, \theta_2)$ l'objectiu de l'apartat del punt de canvi és determinar quines observacions provenen del model amb θ_1, quines del model amb θ_2, i sobretot determinar en quin moment r s'ha produït el canvi.

En aquest cas, el model estadístic és el $Binomial(\theta, m)$, l'usuari haurà d'especificar les dues distribucions a priori del model, $Binomial(\theta_1, m_i)$ i $Binomial(\theta_2, m_i)$ i la distribució a priori del valor del punt de canvi. Seguidament haurà d'introduir les dades pertinents, $\{y_1, \dots, y_n\}$ i $\{m_1, \dots, m_n\}$, per tal de visualitzar les distribucions a posteriori dels paràmetres i la estimació del punt de canvi.

6.5.1 Aparença

En la part de *Configuration Options* s'implementaran els elements per determinar les dues distribucions a priori dels paràmetres θ. Ja que es tracta d'un model $Binomial$ s'utilitzarà la distribució a priori conjugada, és a dir, una distribució $Beta$.

Seguidament mitjançant un quadre per escollir opcions es preguntarà a l'usuari si té coneixement a priori sobre on es troba el punt de canvi. Si no es té coneixement a priori sobre el punt de canvi, per defecte s'utilitzarà una distribució $Uniforme\{1,2, \dots, n\}$, on n és la mida de la mostra. En cas contrari l'usuari haurà d'introduir mitjançant un desplegable, *selectInput,* una variable amb les probabilitats corresponents a cada valor. S'escull aquesta opció ja que els valors a priori més probables poden ser molt distants entre sí i d'aquesta forma l'usuari podrà determinar la distribució lliurement.

Finalment les dades introduïdes seran els valors y de les distribucions $Binomial$ conjuntament amb el seu paràmetre m, que pot ser diferent per a cada valor y. Així doncs s'implementaran dos desplegables.

En la part de *Results*, concretament a la pestanya de *Estimation* es representaran les distribucions a priori i a posteriori dels paràmetres de probabilitat θ_1i θ_2 , tal com es pot observar en la Figura 6.20.

En la pestanya *Change Point,* tal com mostra la Figura 6.21, es troba la distribució a priori del punt de canvi, un missatge d'avís que adverteix que el procés pot tardar uns instants i la distribució a posteriori del punt de canvi.

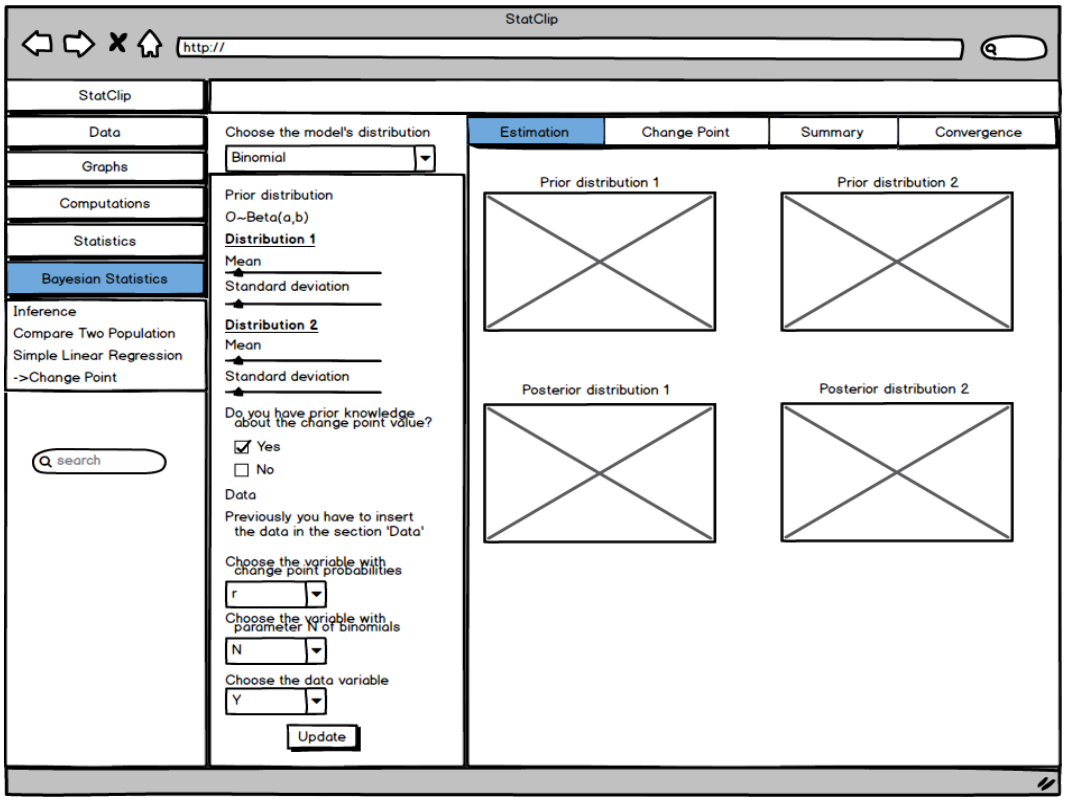


Figura 6.20 Aparença de l'apartat *Change Point* en la pestanya *Estimation*

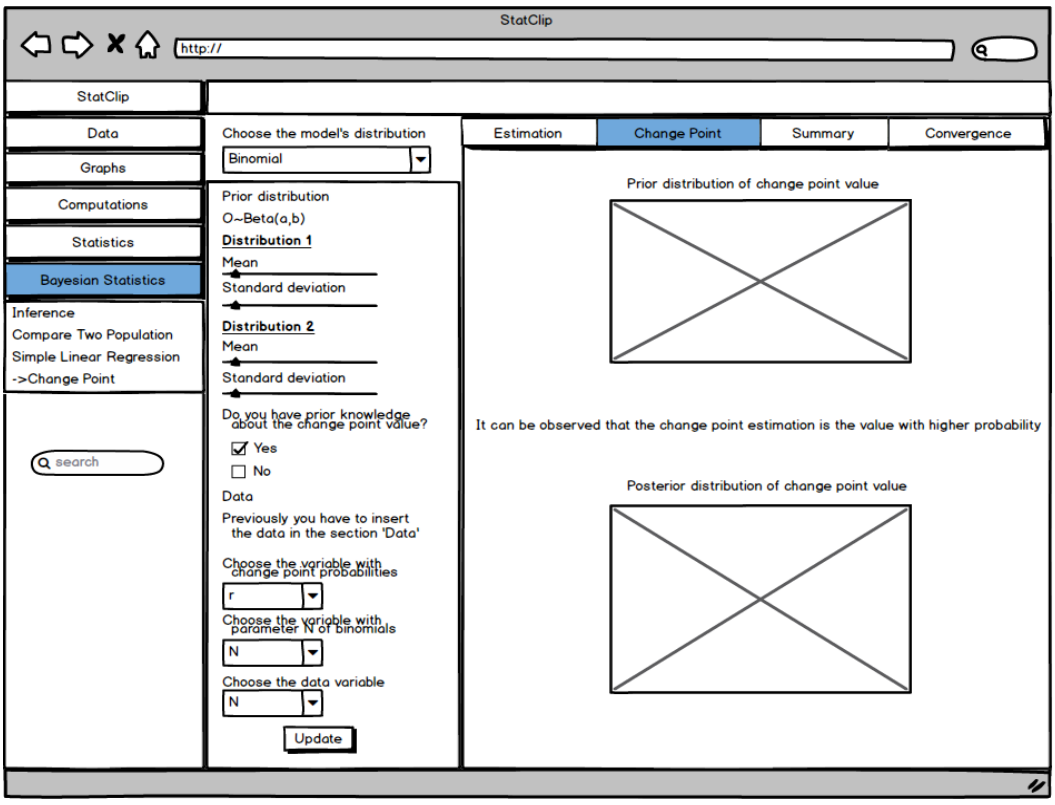


Figura 6.21 Aparença de l'apartat *Change Point* en la pestanya *Change Point*

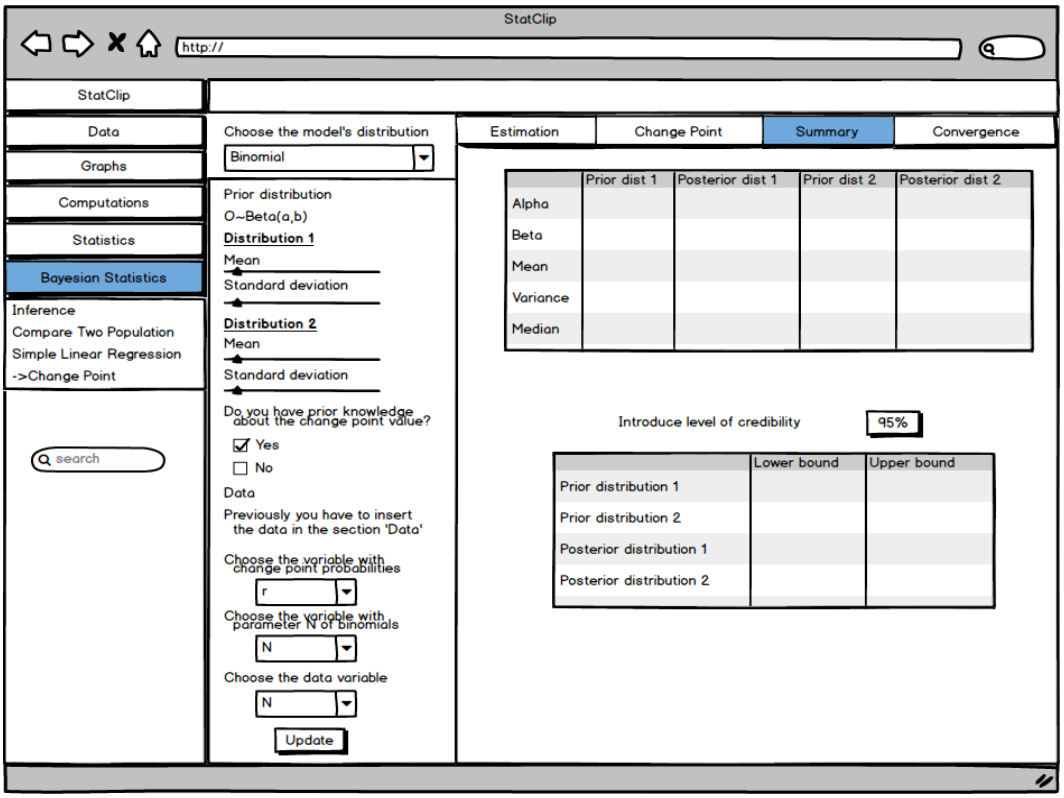


Figura 6.22 Aparença de l'apartat *Change Point* en la pestanya *Summary*

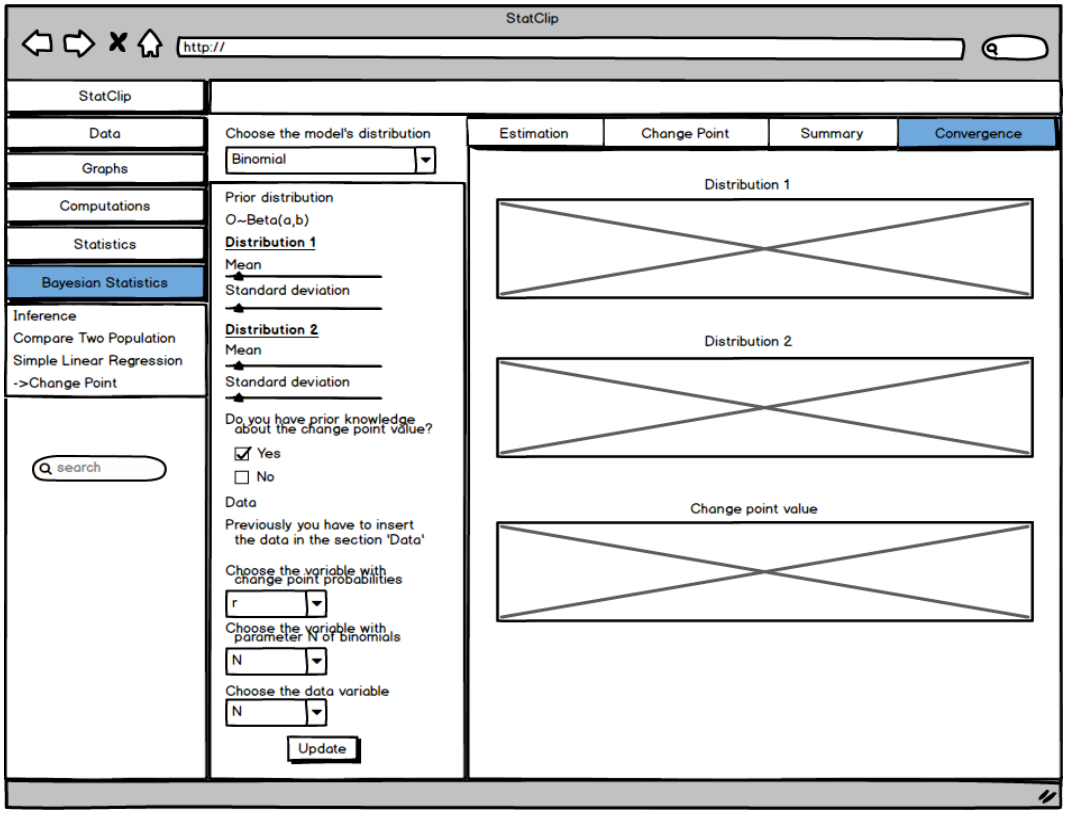


Figura 6.23 Aparença de l'apartat *Change Point* en la pestanya *Convergence*

En la part *Summary* es troben les estimacions puntuals i per interval de les distribucions tal com es pot observar en la Figura 6.22. Finalment a la pestanya *Convergence,* tal com mostra la Figura 6.23, es representa la convergència de la cadena per les distribucions de θ_1, θ_2 i r.

6.5.2 Implementació

La implementació de *Configuration Options* s'elabora anàlogament als altres apartats. Cal destacar que la selecció de la variable on es troben les probabilitats del punt de canvi només apareixerà si l'usuari té coneixement a priori sobre aquesta.

L'algoritme *Gibbs Sampling* haurà de ser utilitzat més d'una vegada per mostrar diferents *outputs*. Per aquest motiu aquest serà implementat dins d'una funció *reactive*. Així doncs, a partir de la distribució conjunta

$$y_i, \dots, y_n|\theta_1, \theta_2, r \sim \prod_{i=1}^{r} P(y_i|\theta_1) \prod_{i=1+r}^{n} P(y_i|\theta_2),$$

on $y_i \sim Binomial(m_i, \theta_1), i = 1, \dots, r$ i $y_i \sim Binomial(m_i, \theta_2), i = r+1, \dots, n$, aleshores les distribucions a priori dels paràmetres de probabilitat són $\theta_1 \sim Beta(a_1, b_1)$ i $\theta_2 \sim Beta(a_2, b_2)$. El punt de canvi pot seguir una distribució $Uniforme\{1,2, \dots, n\}$ si no es té coneixement a priori d'aquest o una distribució discreta, $\pi(r)$. Aleshores la funció de versemblança és:

$$\begin{aligned} L(\theta_1, \theta_2, r; y, m) &\propto \prod_{i=1}^{r} \theta_1^{y_i}(1-\theta_1)^{m_i - y_i} \prod_{i=r+1}^{n} \theta_2^{y_i}(1-\theta_2)^{m_i - y_i} \\ &\propto \theta_1^{\sum_{i=1}^{r} y_i}(1-\theta_1)^{\sum_{i=1}^{r} m_i - \sum_{i=1}^{r} y_i} \theta_2^{\sum_{i=r+1}^{n} y_i}(1-\theta_2)^{\sum_{i=r+1}^{n} m_i - \sum_{i=r+1}^{n} y_i}, \end{aligned}$$

i la distribució a priori és:

$$\pi(\theta_1, \theta_2, r) \propto \theta_1^{a_1 - 1}(1-\theta_1)^{b_1 - 1} \theta_2^{a_1 - 1}(1-\theta_2)^{b_1 - 1} \pi(r)\, I_{(0,1)}(\theta_1) I_{(0,1)}(\theta_2) I_{\{1,\dots,n\}}(r).$$

Amb la versemblança i la distribució a priori s'obté la funció a posteriori de la forma:

$$\begin{aligned} \pi(\theta_1, \theta_2, r|\, y, m) &\propto L(\theta_1, \theta_2, r; y, m)\pi(\theta_1, \theta_2, r) \\ &\propto \theta_1^{\sum_{i=1}^{r} y_i}(1-\theta_1)^{\sum_{i=1}^{r} m_i - \sum_{i=1}^{r} y_i} \theta_2^{\sum_{i=r+1}^{n} y_i}(1-\theta_2)^{\sum_{i=r+1}^{n} m_i - \sum_{i=r+1}^{n} y_i} \theta_1^{a_1 - 1}(1 \\ &- \theta_1)^{b_1 - 1} \theta_2^{a_2 - 1}(1-\theta_2)^{b_2 - 1} \pi(r)\, I_{(0,1)}(\theta_1) I_{(0,1)}(\theta_2) I_{\{1,\dots,n\}}(r) \\ &\propto \theta_1^{\sum_{i=1}^{r} y_i + a_1 - 1}(1-\theta_1)^{\sum_{i=1}^{r} m_i - \sum_{i=1}^{r} y_i + b_1 - 1} \theta_2^{\sum_{i=r+1}^{n} y_i + a_2 - 1}(1 \\ &- \theta_2)^{\sum_{i=r+1}^{n} m_i - \sum_{i=r+1}^{n} y_i + b_2 - 1} \pi(r)\, I_{(0,1)}(\theta_1) I_{(0,1)}(\theta_2) I_{\{1,\dots,n\}}(r). \end{aligned}$$

Per tal d'aproximar els valors esperats de la distribució a posteriori,

$$E[\theta_1, \theta_2, r1|y, m] = \begin{bmatrix} E[\theta_1|y, m] \\ E[\theta_2|y, m] \\ E[r|y, m] \end{bmatrix}$$

,mitjançant el *Gibbs Sampling*, és necessari calcular les distribucions condicionades. La distribució de θ_1 condicionada a θ_2, r, y i m és

$$\begin{aligned}\pi(\theta_1|\theta_2, r, y, m) &\propto \pi(\theta_1, \theta_2, r|\, y, m) \\ &\propto \theta_1^{\sum_{i=1}^{r} y_i + a_1 - 1}(1-\theta_1)^{\sum_{i=1}^{r} m_i - \sum_{i=1}^{r} y_i + b_1 - 1}\theta_2^{\sum_{i=r+1}^{n} y_i + a_2 - 1}(1 \\ &- \theta_2)^{\sum_{i=r+1}^{n} m_i - \sum_{i=r+1}^{n} y_i + b_2 - 1}\pi(r)\, I_{(0,1)}(\theta_1) I_{(0,1)}(\theta_2) I_{\{1,\ldots,n\}}(r) \\ &\propto \theta_1^{\sum_{i=1}^{r} y_i + a_1 - 1}(1-\theta_1)^{\sum_{i=1}^{r} m_i - \sum_{i=1}^{r} y_i + b_1 - 1}\, I_{(0,1)}(\theta_1)\end{aligned}$$

i aleshores

$$\theta_1 \sim Beta\left(a_1 + \sum_{i=1}^{r} y_i\,,\ \ b_1 + \sum_{i=1}^{r} m_i - \sum_{i=1}^{r} y_i\right).$$

La distribució de θ_2 condicionada a θ_1, r, y i m és:

$$\begin{aligned}\pi(\theta_2|\theta_1, r, y, m) &\propto \pi(\theta_1, \theta_2, r|\, y, m) \\ &\propto \theta_1^{\sum_{i=1}^{r} y_i + a_1 - 1}(1-\theta_1)^{\sum_{i=1}^{r} m_i - \sum_{i=1}^{r} y_i + b_1 - 1}\theta_2^{\sum_{i=r+1}^{n} y_i + a_2 - 1}(1 \\ &- \theta_2)^{\sum_{i=r+1}^{n} m_i - \sum_{i=r+1}^{n} y_i + b_2 - 1}\pi(r)\, I_{(0,1)}(\theta_1) I_{(0,1)}(\theta_2) I_{\{1,\ldots,n\}}(r) \\ &\propto \theta_2^{\sum_{i=r+1}^{n} y_i + a_2 - 1}(1-\theta_2)^{\sum_{i=r+1}^{n} m_i - \sum_{i=r+1}^{n} y_i + b_2 - 1} I_{(0,1)}(\theta_2)\end{aligned}$$

i aleshores

$$\theta_2 \sim Beta\left(\mathrm{a}_2 + \sum_{i=\mathrm{r}+1}^{n} y_i\,,\ \ \mathrm{b}_2 \sum_{i=\mathrm{r}+1}^{n} m_i - \sum_{i=\mathrm{r}+1}^{n} y_i\right).$$

La distribució de r condicionada a θ_1, θ_2, y i m és:

$$\pi(r = k|\theta_1, \theta_2, y, m)$$
$$= \frac{\left(\theta_1^{\sum_{i=1}^{k} y_i + a_1 - 1}(1-\theta_1)^{\sum_{i=1}^{k} m_i - \sum_{i=1}^{k} y_i + b_1 - 1}\theta_2^{\sum_{i=k+1}^{n} y_i + a_2 - 1}(1-\theta_2)^{\sum_{i=k+1}^{n} m_i - \sum_{i=k+1}^{n} y_i + b_2 - 1}\right)\pi_r(k)}{\left(\sum_{s=1}^{n} \theta_1^{\sum_{i=1}^{s} y_i + a_1 - 1}(1-\theta_1)^{\sum_{i=1}^{s} m_i - \sum_{i=1}^{s} y_i + b_1 - 1}\theta_2^{\sum_{i=s+1}^{n} y_i + a_2 - 1}(1-\theta_2)^{\sum_{i=s+1}^{n} m_i - \sum_{i=s+1}^{n} y_i + b_2 - 1}\,\pi_r(s)\right)}.$$

Ja que el càlcul de $P(r = k|\theta_1, \theta_2, y, m)$ és extens, s'implementa una funció pel càlcul del numerador i una altra pel quocient, per tal d'agilitzar el procés en el fitxer *ChangePointFunctions.R*.

Així doncs en la funció *reactive* s'implementa tot l'algoritme *Gibbs Sampling.*

A l'observar que l'algoritme no era immediat es volia implementar una forma de conèixer el temps restant. Així doncs amb la funció *setWinProgressBar* s'obre una finestra emergent amb una barra de progrés en % que es tanca al finalitzar les iteracions. Aquesta funció només és vàlida quan l'usuari executa l'aplicació amb programa *R* des d'un ordinador local. Ja que al penjar-lo en el servidor de *Shiny* no funcionarà es decideix deixar-la com a comentari.

Seguidament en la implementació de la part *Estimation* les distribucions predictives a priori son anàlogues a la pestanya de *Estimation* de *Compare two population.*

Per la representació de les distribucions predictives a posteriori es crida la funció on es troba l'algoritme i es representen les simulacions de θ_1i θ_2.

En la part *Change Point*, concretament en la representació de la distribució a priori de r, si l'usuari té informació, mitjançant un *selectInput,* un desplegable, es llegirà la variable i es representarà. En cas contrari, per defecte es representarà una distribució $Uniforme\{1,2,\dots,n\}$.

Seguidament per representar la distribució a posteriori del punt de canvi es crida la funció *reactive,* on es troba l'algoritme *Gibbs Sampling,* i es representen les simulacions de r. En el gràfic s'observarà que, en certes ocasions, no tots els valors de r tenen probabilitat de ser el punt de canvi i per tant l'*R* no els representarà. Així doncs el punt de canvi estimat serà el valor amb la probabilitat més gran.

En la pestanya de *Summary* s'implementen les taules, amb les estimacions puntuals i per interval, anàlogament als altres apartats.

Finalment en la part *Convergence* es representa la convergència de les simulacions obtingudes de l'algoritme per θ_1 ,θ_2 i r.

6.6 *BayesClip* a la web

Un cop finalitzada la creació de l'aplicació es procedeix a penjar-la en un servidor web.

El servidor web escollit és el de *shinyapps* ja que té més avantatges que un servidor web normal **[15]**. El servidor de *shinyapps* ja consta de tots els paquets de R instal·lats pel correcte funcionament de totes les aplicacions, a més a més, permet que molts usuaris puguin utilitzar l'aplicació a la vegada.

Mitjançant el compte d'usuari del tutor d'aquest treball de fi de grau, Lluís Marco, es penja l'aplicació a l'adreça: https://noctilabium.shinyapps.io/BayesClip/.

7. Avaluació de l'impacte de *BayesClip*

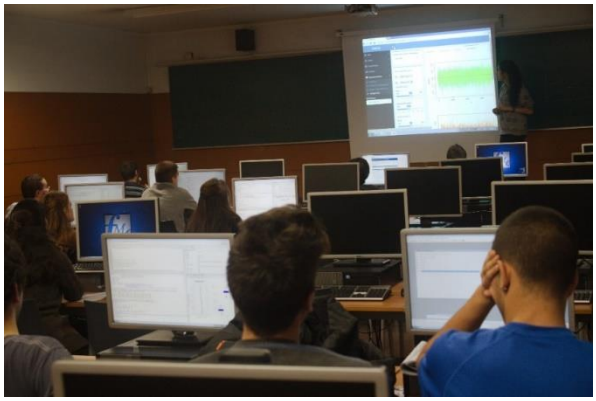
Un cop finalitzada l'aplicació i penjada en el servidor corresponent es decideix testar-la en usuaris per avaluar l'impacte que té.

La mostra seleccionada són els alumnes del Grau d'Estadística que cursen l'assignatura de Mètodes Bayesians en el curs 2015-2016.

El dia del test serà l'últim dia de classe lectiva ja que d'aquesta forma els usuaris ja hauran adquirit els coneixements bàsics de l'estadística Bayesiana i només serà necessari explicar el funcionament de l'aplicació.

Per avaluar l'impacte de l'aplicació i determinar si és útil i intuïtiva es decideix que els alumnes resolguin uns exercicis mitjançant l'aplicació i responguin una petita enquesta

Així doncs, abans de lliurar els exercicis als alumnes se'ls hi va explicar què és el *Shiny* i com funciona. Seguidament, tal com mostra la Imatge 7.1, se'ls hi va explicar l'estimació del punt de canvi mitjançant un exemple amb diverses probabilitats. Ja que és un concepte que no van estudiar en profunditat durant el curs,



Imatge 7.1 Presentació als alumnes

7.1 Exercicis resolts

Per tal de que els alumnes no estiguin un temps excessiu elaborant els exercicis es decideix escollir-ne uns que ja han resolt i treballat a classe durant el curs. D'aquesta forma s'avaluarà l'agilitat en utilitzar l'aplicació i no en resoldre l'exercici en sí.

S'escullen tres exercicis. El primer referent a l'estimació d'una freqüència, és a dir mitjançant un model estadístic $Poisson$. El segon referent a l'estimació de la diferència de dues

proporcions, mitjançant un model estadístic $Binomial$. Finalment el tercer, tracta sobre l'estimació d'un model de regressió lineal simple.

Seguidament es mostren els exercicis plantejats conjuntament amb l'explicació detallada i la seva resolució mitjançant l'aplicació.

7.1.1 Exercici 1: Estimació d'una freqüència

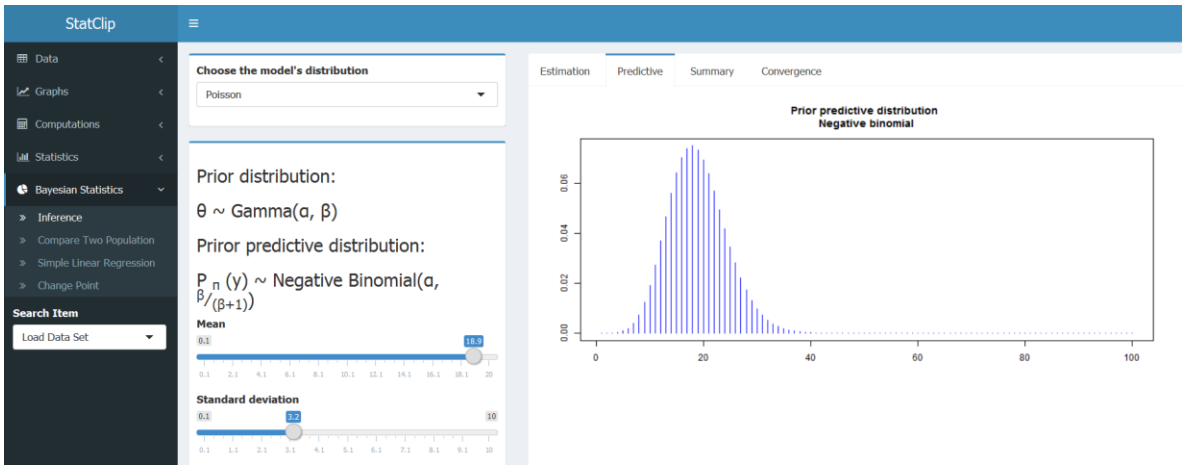
El portal d'internet de l'associació cultural La Sèpia Verda desconeix la freqüència de visitants setmanals al portal. Amb l'objectiu de sortir de la ignorància s'ha recollit el nombre de visitants de les darreres 10 setmanes. Les dades es troben a l'arxiu *sepiaverda.txt*. Els responsables de l'associació viuen en el convenciment de que rarament el nombre de visitants serà inferior a 5 i rarament superior a 40.

De l'enunciat, es determina que el model és $M = \{Poisson(\theta), \theta \in \mathbb{R}^+\}$.

a) Tria els paràmetres de la distribució a priori (Nota: Pot ser útil triar els paràmetres de la a priori utilitzant la predictiva a priori).

A partir del model estadístic $Poisson$ s'utilitza el model conjugat per determinar la distribució a priori del paràmetre. Aleshores la distribució a priori és $\pi(\theta) = Gamma(a, b)$. Per determinar els paràmetres de la $Gamma$ s'utilitzarà la distribució predictiva a priori, $NegativaBinomial$, ja que la informació que aporta l'enunciat fa referència a les dades i no al paràmetre.

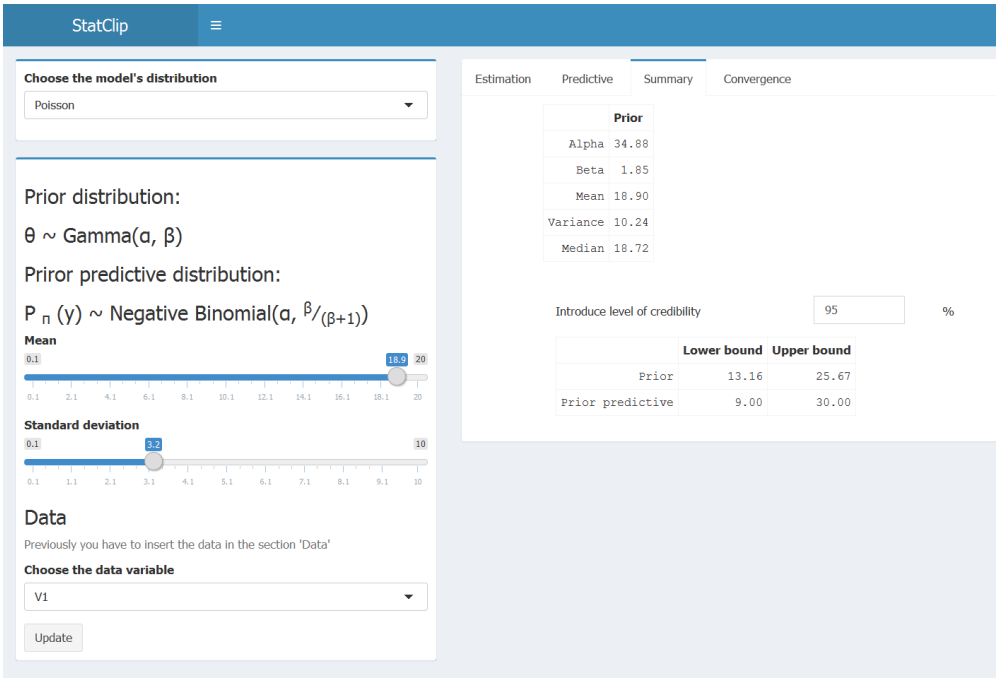
Mitjançant *BayesClip* es determina la distribució predictiva a priori de tal forma que la major part de la probabilitat es trobi entre 5 i 40 tal com mostra la Imatge 7.2.



Imatge 7.2 Distribució predictiva a priori del model *Poisson*

Un cop determinada la distribució predictiva a priori mitjançant les barres lliscants de l'esperança i la desviació tipus de la distribució a priori, s'observen els paràmetres de la distribució a priori a la taula resum que apareix a la pestanya *Summary*. En aquest cas, tal

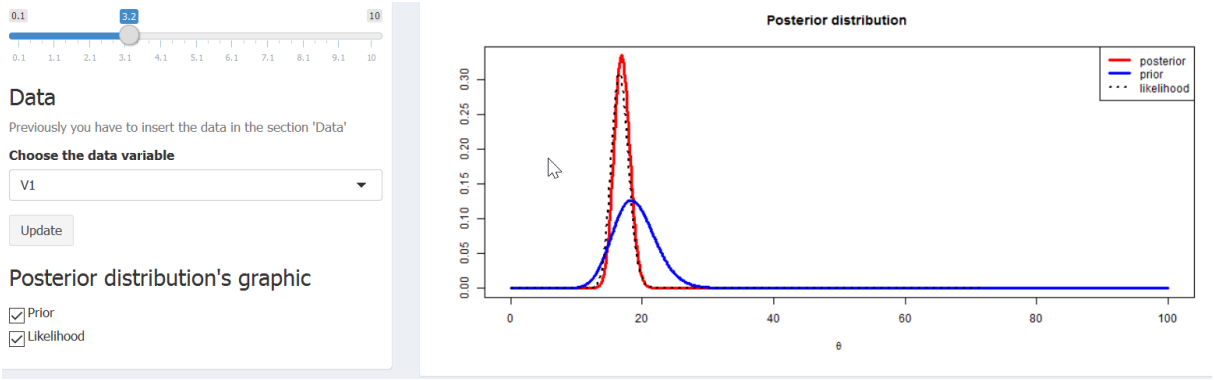
com mostra la Imatge 7.3, els paràmetres d'una possible distribució a priori són $a = 34.88$ i $b = 1.85$, determinant així $\pi(\theta) = Gamma(34.88, 1.85)$.



Imatge 7.3 Pestanya *Summary* del model estadístic *Poisson*

b) Dibuixa la distribució a priori i la versemblança en un mateix gràfic.

Un cop introduïdes les dades en l'apartat *Data* de l'aplicació, es selecciona la variable a l'apartat *Inference* i es pressiona el botó *update* per obtenir el gràfic de la distribució a posteriori, conjuntament amb la distribució a priori i la versemblança.



Imatge 7.4 Representació de la distribució a posteriori conjuntament amb l'esperança i la variança del model *Poisson*

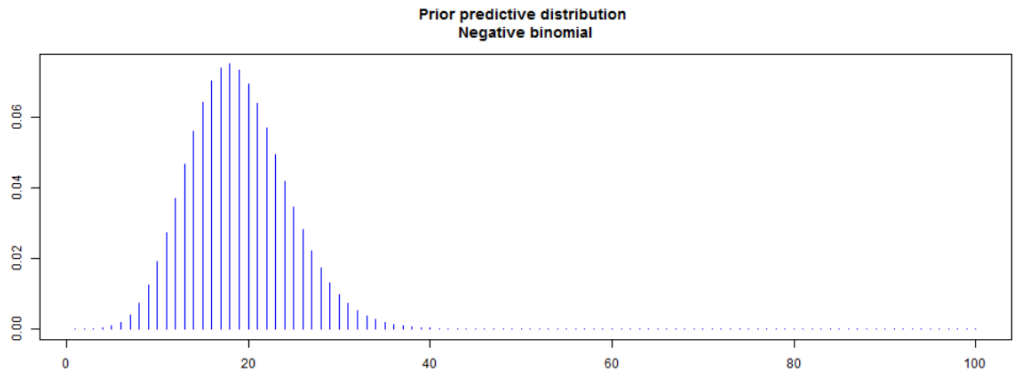
En el gràfic de la Imatge 7.4 s'observa que la distribució a posteriori i la versemblança són molt semblants. El principi de versemblança diu que tota la informació que tenen les dades sobre el paràmetre (en aquest cas θ) està a la funció de versemblança. Per tant, com més

gran és la versemblança més versemblant és que θ sigui el verdader paràmetre de la població. Així doncs, observant el gràfic de la Imatge 7.4, es determina que és més versemblant que el paràmetre θ sigui igual a 17.

c) Dibuixa la predictiva a priori.

Tal com s'ha esmentat en l'apartat a), el gràfic de la distribució predictiva a priori es troba en la pestanya *Predictive*. Aquest reflecteix la informació que ens aporta l'enunciat, que és que rarament el nombre de visitants serà inferior a 5 i rarament superior a 40.

En aquest cas al ser un model conjugat la distribució predictiva és exacta i no és necessària la simulació.

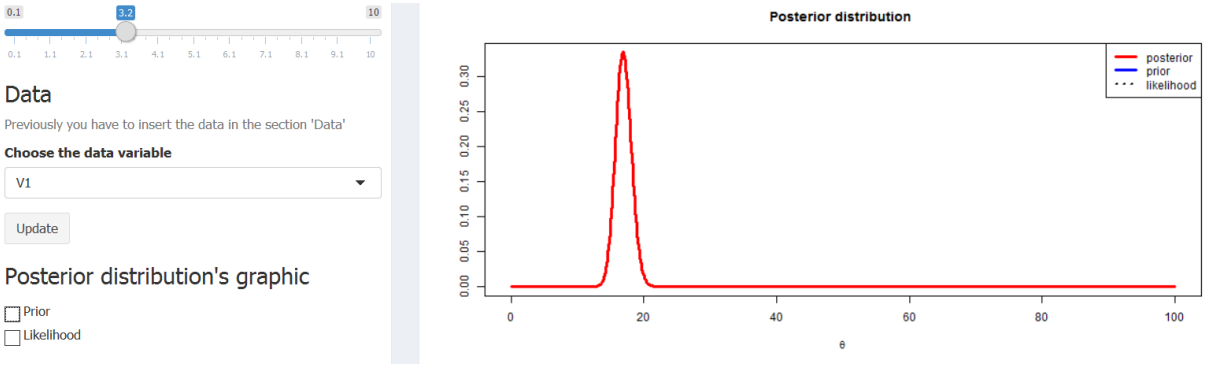


Imatge 7.5 Gràfic de la distribució predictiva a priori del model *Poisson*

d) Dibuixa la distribució a posteriori, i dóna un interval de credibilitat del 95%.

Per dibuixar solament la distribució a posteriori, un cop introduïdes les dades i pressionat el botó *update*, s'utilitzen les opcions *Posterior distribution's graphic* que s'observen en la Imatge 7.6.

Un cop dibuixada la distribució a posteriori s'accedeix a la pestanya *Summary,* es determina el nivell de credibilitat de l'interval i aquest es calcula automàticament. En aquest cas, observant la Imatge 7.7, l'interval de credibilitat de la distribució a posteriori és $[14.69, 19.38]$.



Imatge 7.6 Gràfic de la distribució a posteriori del model *Poisson*

Introduce level of credibility 95 %

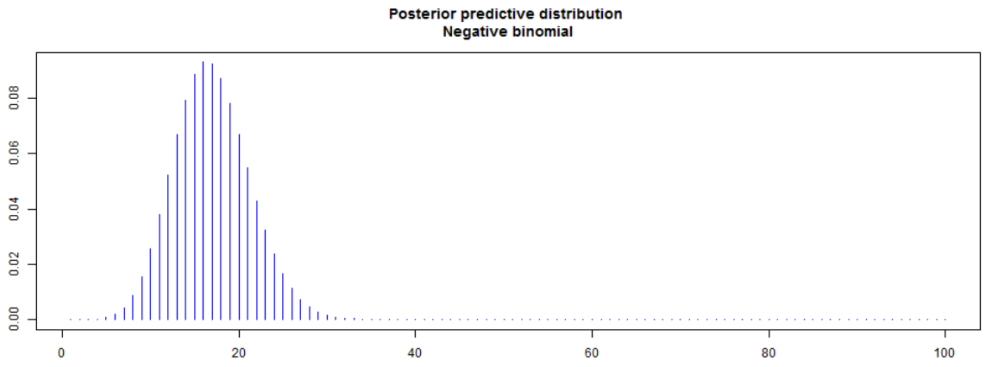
	Lower bound	Upper bound
Prior	13.16	25.67
Prior predictive	9.00	30.00
Posterior	14.69	19.38
Posterior predictive	9.00	26.00

Imatge 7.7 Interval de credibilitat del 95% per la distribució a posteriori

e) Dibuixa la distribució predictiva a posteriori, i dóna un interval de credibilitat del 90%.

Per tal de dibuixar la distribució predictiva a posteriori s'han d'haver introduït les dades a l'apartat *Data* i s'ha d'haver seleccionat la variable en l'apartat *Inference*. Després, un cop pressionat el botó *update*, s'observa el gràfic de la Imatge 7.8, on hi apareix la distribució predictiva a posteriori. Per calcular l'interval de credibilitat d'aquesta distribució, s'indica el nivell de credibilitat en la pestanya *Summary* i s'obté la taula de la Imatge 7.9.Imatge 7.7

La distribució predictiva a posteriori reflecteix tot el coneixement de les dades futures i per tant en aquest cas es determina que les dades futures es mouran amb una credibilitat del 90% en l'interval $[10, 24]$.



Imatge 7.8 Gràfic de la distribució predictiva a posteriori del model *Poisson*

Introduce level of credibility 90 %

	Lower bound	Upper bound
Prior	13.96	24.45
Prior predictive	11.00	28.00
Posterior	15.04	18.97
Posterior predictive	10.00	24.00

Imatge 7.9 Interval de credibilitat del 90% per la distribució predictiva a posteriori

7.1.2 Exercici 2: Estimació de la diferència de dues proporcions

Ens trobem davant d'un assaig clínic per valorar si els pacients afectats per cremades hipodèrmiques es recuperen més ràpidament quan el tractament combina certa crema antisèptica amb un apòsit hidrocoloide que quan utilitza solament la crema antisèptica. Es coneix amb força certesa que només aproximadament el 60% dels pacients es recupera amb aquest darrer recurs. L'equip investigador té, per altre banda, motius teòrics i indicis empírics sorgits del treball quotidià d'infermeria que fan pensar amb força optimisme que el tractament combinat és més efectiu que el tractament simple. S'ha organitzat un experiment amb n pacients, $n/2$ dels quals s'escullen aleatòriament per ser atesos amb el tractament innovador (combinació de crema antisèptica i apòsit hidrocoloide), en tant que els $n/2$ restants se'ls aplicarà el tractament convencional (només crema). Les probabilitats a priori es defineixen sota la convicció anticipada de que *Pc* (tractament convencional) es troba quasi amb seguretat entre 0.4 i 0.8, amb alta probabilitat en un veïnatge de 0.6, i que es redueix ràpidament quan s'allunya d'aquest punt, i en quant a *Pe* (tractament experimental), es troba en un entorn de 0.8, amb minsa probabilitat d'estar fora de l'interval $[0.7, 0.9]$.

A la següent taula trobem la distribució d'una mostra de 80 pacients segons el tractament assignat i segons si es recuperen o no:

Tractament	Sí	No	Total
Experimental	30	10	40
Convencional	24	16	40
Total	54	26	80

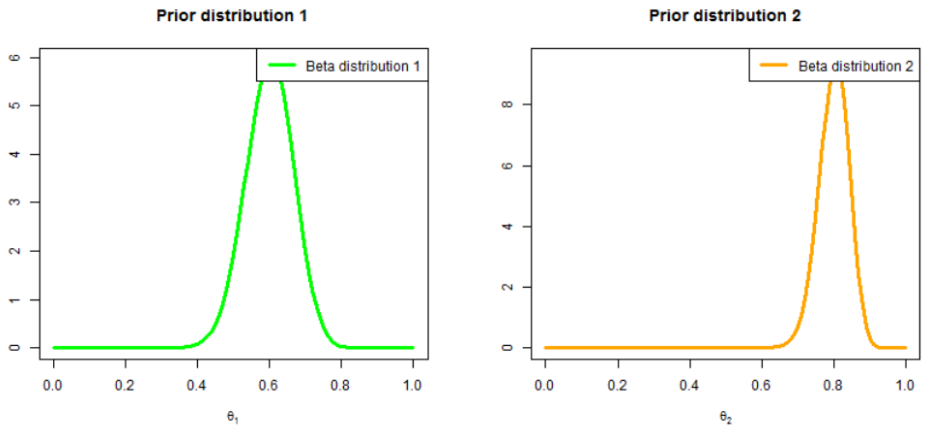
De l'enunciat, es determina que el model és $M = \{Binomial(\theta, m),\ \theta \in (0,1)\}$.

a) Escull les distribucions a priori d'acord amb l'enunciat.

Es determina:

- *Population 1* → Tractament convencional.
- *Population 2* → Tractament experimental.

Aleshores per determinar la distribució a priori del tractament 1 es dibuixa una distribució on la major part de la probabilitat es trobi en el valor 0.6, ja que l'enunciat informa que aproximadament el 60% dels pacients es recupera amb aquest tractament. Per determinar la distribució a priori del tractament 2 es dibuixa la distribució de forma que la major part de la probabilitat es trobi en un entorn de 0.8, amb la minsa probabilitat d'estar fora de l'interval $[0.7, 0.9]$, tal com es mostra en la Imatge 7.10.



Imatge 7.10 Distribucions a priori del tractament convencional i del tractament experimental

Un cop determinades les distribucions a priori visualment s'observa la taula resum d'aquestes en l'apartat de *Summary*. Així doncs es determina que la distribució a priori pel tractament provisional és $\pi(\theta_1) = Beta(32.46, 21.64)$ i la distribució a priori pel tractament experimental és $\pi(\theta_2) = Beta(71.76, 17.94)$, tal com mostra la Imatge 7.11.

Population 1

	Prior
Alpha	32.46
Beta	21.64
Mean	0.60
Variance	0.00
Median	0.60

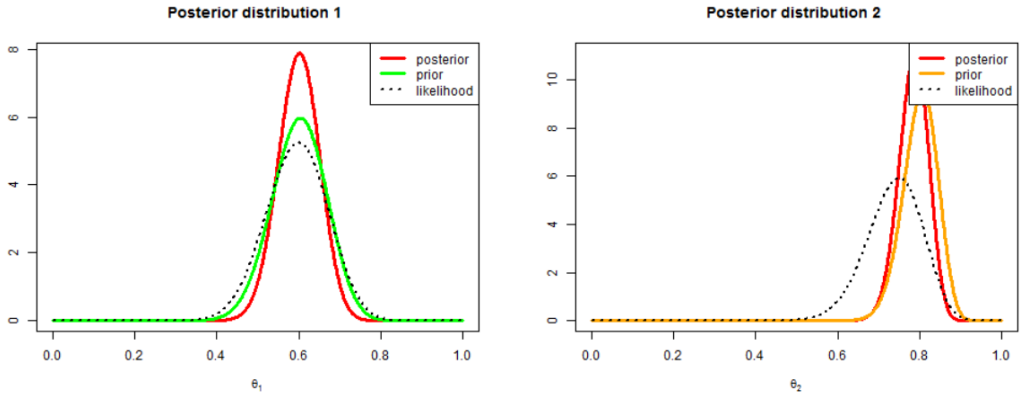
Population 2

	Prior
Alpha	71.76
Beta	17.94
Mean	0.80
Variance	0.00
Median	0.80

Imatge 7.11 Taula resum de les distribucions a priori del tractament convencional i del tractament experimental

b) Dibuixa conjuntament les distribucions a priori, a posteriori i les versemblances.

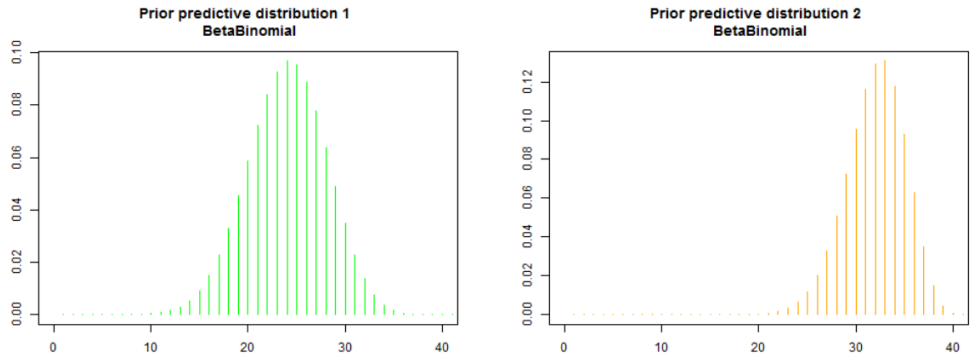
Per dibuixar les distribucions a posteriori s'han d'haver introduït les dades i pressionat el botó *update*. En la Imatge 7.12 es mostren els gràfics de les distribucions a priori, a posteriori i les versemblances del tractament convencional i del tractament experimental. S'observa que per el tractament convencional la distribució a posteriori és molt més alta que la versemblança, indicant d'aquesta forma que la distribució a posteriori transmet molta informació ja que conté tota la informació que aporten les dades més la informació a priori dels experts. En gràfic del tractament experimental s'observa que la versemblança es troba desplaçada de la distribució a posteriori, indicant que és més versemblant que el tractament experimental no tingui una taxa tant alta de pacients que es recuperen ràpidament.



Imatge 7.12 Gràfics de les distribucions a priori, a posteriori i les versemblances del tractament convencional i del tractament experimental

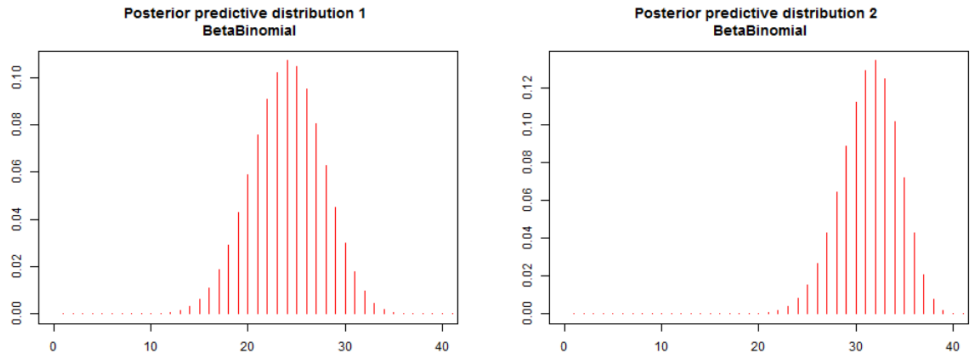
c) Dibuixa la predictiva a priori i a posteriori per a cadascun dels tractaments.

Per tal d'observar les distribucions predictives a priori que es mostren en la Imatge 7.13 s'ha d'anar a la pestanya *Predictive*. Les distribucions predictives a priori dels tractaments mostren les prediccions futures abans d'haver observat les dades.



Imatge 7.13 Gràfic de les distribucions predictives a priori del tractament convencional i del tractament experimental

Un cop observades les dades i actualitzat el model Bayesià s'obtenen les distribucions predictives a posteriori de la Imatge 7.14. En aquesta s'observa el coneixement de les prediccions futures. Si es comparen les distribucions predictives a priori amb les distribucions predictives a posteriori no s'observa diferència, això és degut a que els coneixements a priori coincideixen amb la informació que aporten les dades.

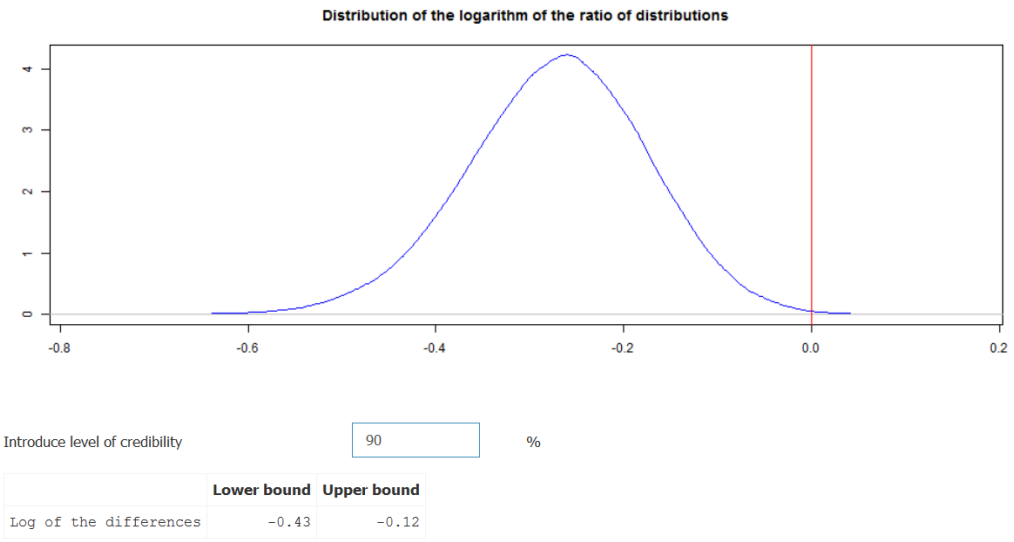


Imatge 7.14 Gràfic de les distribucions predictives a posteriori del tractament convencional i del tractament experimental

d) Dibuixa la distribució a posteriori del logaritme de la ràtio de les diferències. Dóna un interval de credibilitat del 90% i interpreta el resultat.

Per tal d'observar la distribució a posteriori del logaritme de la ràtio de les diferències conjuntament amb l'interval de credibilitat s'ha d'anar a la pestanya *Difference*. El gràfic de la Imatge 7.15 determina que la diferència entre tractaments és significativa amb un nivell de

credibilitat del 90%, ja que tant el gràfic com l'interval no inclouen el valor 0. Al no incloure el 0 es determina que els tractaments són diferents. En el gràfic i l'interval de credibilitat s'observa que el tractament experimental té millors resultats.



Imatge 7.15 Distribució a posteriori del logaritme de la ràtio de les diferències conjuntament amb l'interval de credibilitat del 90%

7.1.3 Exercici 3: Regressió lineal simple

En el fitxer *animals.csv* tenim dades de diferents animals, on tenim el pes del cervell i el pes del cos, i volem veure si a partir del pes del cos podem conèixer el pes del cervell (les dades utilitzades estan en escala logarítmica ja que prèviament s'ha observat que la variabilitat augmenta amb les variables).

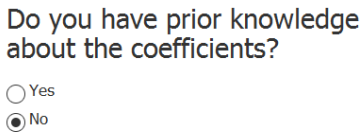
a) Formula el model

Es considera el logaritme del pes del cervell com a variable resposta i el logaritme del pes del cos com a variable explicativa. Aleshores el model de regressió és $y = \beta_0 + \beta_1 x + \varepsilon$.

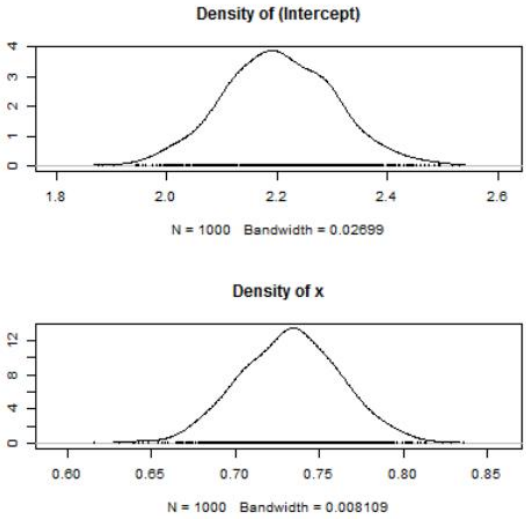
b) Defineix les distribucions a priori per als paràmetres sota el supòsit de no informació i mostra les distribucions a posteriori per als paràmetres.

Per indicar que no es té informació a priori sobre els paràmetres del model s'utilitza el *checkboxgroup* que s'observa en la Imatge 7.16, seleccionant la opció *No*.

Seguidament en la pestanya *Plot* s'observen les funcions de densitat dels paràmetres que es mostren en la Imatge 7.17. Aquestes funcions a posteriori determinen que el paràmetre β_0 es mou entre els valors 2 i 2.4, i el paràmetre $\beta_1$es mou entre els valors 0.65 i 0.8.



Imatge 7.16 Representació de la pregunta: Tens informació a priori sobre els coeficients?



Imatge 7.17 Funcions de densitat dels paràmetres

c) Dóna una estimació puntual per als paràmetres.

En la pestanya *Model* s'observa una taula resum amb les estimacions puntuals dels paràmetres, tal com mostra la Imatge 7.18. En aquest cas l'estimació màxim versemblant pel paràmetre que determina la intersecció és de 2.2 i el paràmetre que determina la pendent és de 0.73.

	Post. mean
Intercept	2.20
x	0.73

Imatge 7.18 Taula resum de les estimacions dels paràmetres del model de regressió lineal simple

7.1.4 Avaluació exercicis alumnes

Després de que els alumnes lliuressin les respostes dels exercicis en un arxiu *.doc* es procedeix a analitzar els resultats.

Un total de 14 alumnes van resoldre els exercicis. D'aquests 14 documents, tres estan repetits. Una possible explicació és que 3 dels alumnes van elaborar els exercicis conjuntament en un mateix ordinador.

Malgrat el poc temps que van tenir els alumnes per elaborar els exercicis dels 14 tots ells van elaborar sencer l'exercici 1, una persona va fer l'exercici 2, i 5 l'exercici 3.

De l'exercici 1, tothom va resoldre tots els apartats. I els pocs errors trobats són referents a conceptes de teoria i no de com utilitzar l'aplicació.

De l'exercici 2, l'única persona que el va contestar, té un petit error de concepte entre paràmetres d'una distribució i l'esperança d'aquesta. L'exercici no està complert, ja que manca la resposta del últim apartat.

De l'exercici 3, una persona l'elabora perfectament, i els altres quatre el van deixar incomplert, segurament per manca de temps.

Així doncs, després d'analitzar els exercicis resolts i veure com els van elaborar a classe, es conclou que si haguessin tingut més temps molt possiblement els haguessin resolt tots. Els element demanats en els exercicis es troben fàcilment, llevat dels paràmetres de les distribucions. Finalment és conclou que persones amb coneixements d'estadística Bayesiana poden utilitzar l'aplicació perfectament.

7.2 Enquesta

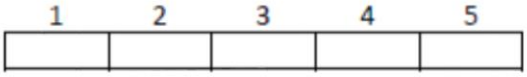
Un cop lliurats els exercicis els alumnes van procedir a contestar l'enquesta (que es troba en la Figura 7.1) on les tres primeres preguntes corresponen a avaluar uns conceptes en una escala de l'1 al 5, essent 1 molt en desacord i 5 molt d'acord, i les 3 últimes preguntes són d'opcions obertes.

Referent a la primera pregunta: Aquesta aplicació serà útil per aprendre els conceptes de l'assignatura "Mètodes Bayesians", s'observa en el Gràfic 7.1 que el 43% dels alumnes sí esta molt d'acord en que aquesta aplicació serà útil per explicar i aprendre conceptes d'estadística Bayesiana. S'observa que aproximadament un 28% dels alumnes no està ni d'acord ni en desacord referent aquesta afirmació i cap alumne hi està en desacord.

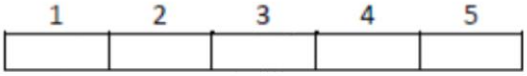
Desenvolupament d'una aplicació per a l'aprenentatge i la docència de l'Estadística Bayesiana

Amb una escala del 1 al 5, essent **1** molt en **desacord** i **5** molt **d'acord**, contesta després d'haver provat la aplicació.

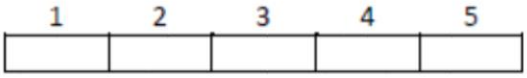
1. Aquesta aplicació serà útil per aprendre els conceptes de l'assignatura "Mètodes Bayesians"



2. Visualment, l'aplicació m'agrada



3. L' aplicació és intuïtiva i fàcil de fer servir



4. Digues les 3 coses que **MÉS** t'han agradat de l'aplicació

➢

➢

➢

5. Digues les 3 coses que **MENYS** t'han agradat de l'aplicació

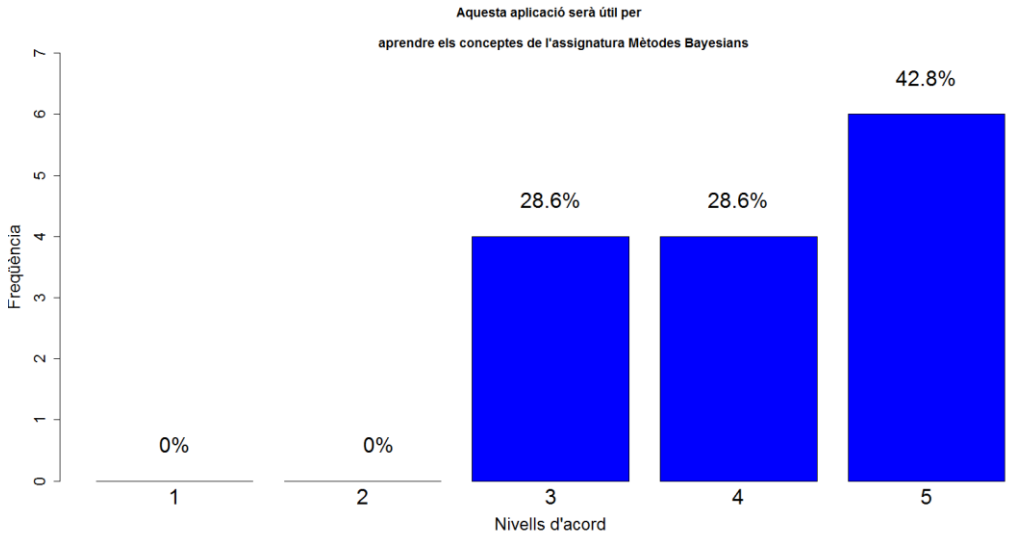
➢

➢

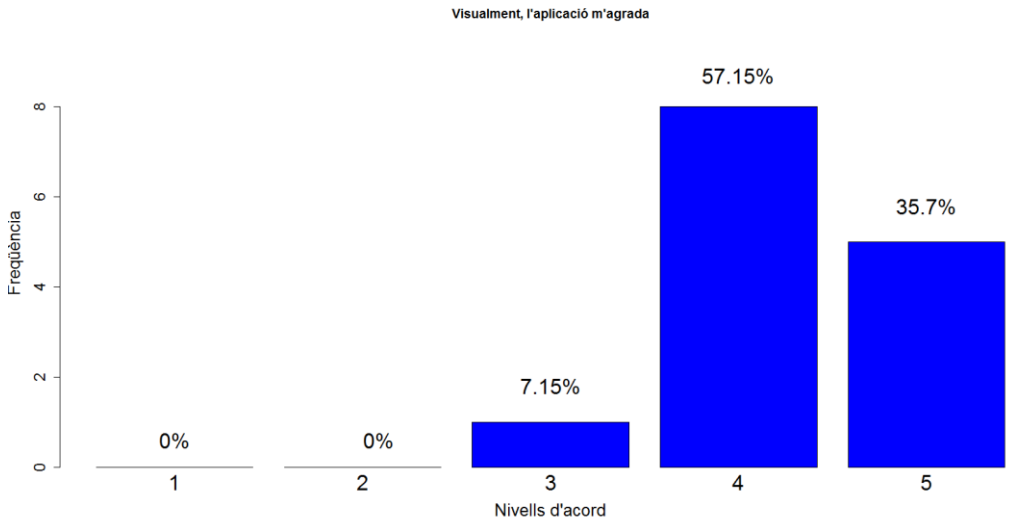
➢

6. Escriu, si vols, qualsevol altre comentari que tinguis.

Figura 7.1 Enquesta lliurada als alumnes



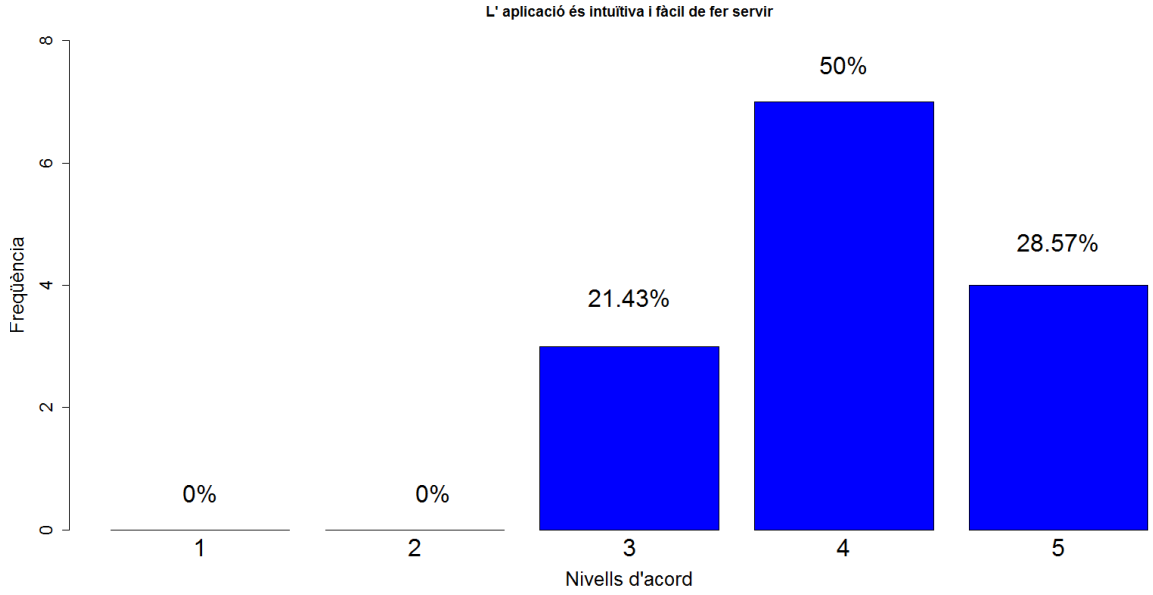
Gràfic 7.1 Resultats qüestionari. Pregunta 1: Valorar de l'1 al 5 si *BayesClip* serà útil per aprendre els conceptes de l'assignatura "Mètodes Bayesians". Essent 1 molt desacord i 5 molt d'acord.



Gràfic 7.2 Resultats qüestionari. Pregunta 2: Valorar de l'1 al 5 si BayesClip visualment agrada. Essent 1 molt desacord i 5 molt d'acord.

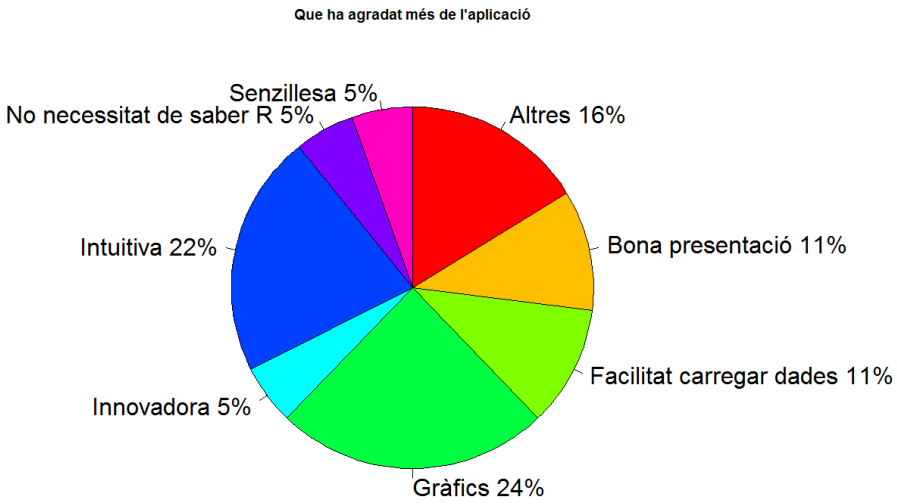
Els resultats de la segona pregunta, que es poden observar en el Gràfic 7.2, determinen que a un 57% dels alumnes visualment els hi agrada l'aplicació i a un 35% els hi agrada molt. Cap alumne hi està en desacord.

En el Gràfic 7.3 s'observen els resultats de la tercera pregunta. Es determina que el 50% dels alumnes creu que l'aplicació sí es intuïtiva i fàcil d'utilitzar, un 28% hi està molt d'acord i cap alumne hi està en desacord.



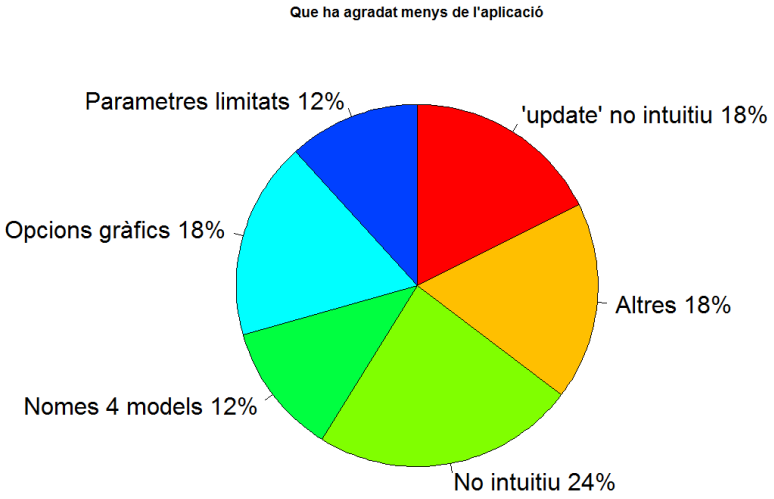
Gràfic 7.3 Resultats qüestionari. Pregunta 3: Valorar de l'1 al 5 si *BayesClip* és intuïtiva i fàcil de fer servir. Essent 1 molt desacord i 5 molt d'acord.

Els resultats de la quarta pregunta, representats en el Gràfic 7.4, determinen que l'aplicació és intuïtiva i que les característiques dels gràfics són bones. L'aplicació també consta de bona presentació i facilitat en carregar les dades.



Gràfic 7.4 Resultats qüestionari. Pregunta 4: dir 3 coses que més t'han agradat de l'aplicació

Els aspectes a millorar de l'aplicació, segons el Gràfic 7.5, són les opcions dels gràfics, la descripció del botó *update* i en general que l'aplicació sigui poc intuïtiva. Els alumnes que diuen que és poc intuïtiva no ho han reflectit de forma clara en la pregunta 3, ja que el 78.6% dels alumnes en aquella pregunta estaven d'acord en que l'aplicació sí és intuïtiva. Als alumnes tampoc els hi ha agradat que només puguin utilitzar 4 models estadístics i que els paràmetres de les distribucions estiguin limitats.



Gràfic 7.5 Resultats qüestionari. Pregunta 5: dir 3 coses que menys t'han agradat de l'aplicació

L'última pregunta, on es demanava qualsevol altre comentari, va ser contestada per 2 persones on esmenten la seva bona opinió sobre l'aplicació. La seva resposta és:

- Ja es podria haver fet abans aquest programa.
- Amb l'estona que hem tingut no hi ha suficient per un bon feedback, però té bona pinta l'aplicació.

Així doncs, es conclou que l'aplicació ha tingut un bon impacte entre els alumnes, ja que ha sigut valorada positivament pels seus gràfics, on s'esmenta la facilitat en representar les distribucions amb diferents valors, per ser intuïtiva, per la facilitat de carregar les dades i la seva bona presentació. Per tal de solucionar els aspectes negatius, en un possible futur, s'intentarà introduir alguns texts explicatius per indicar cada element de l'aplicació i solucionar els problemes d'intuïció amb el botó *update*.

8. Conclusions

El present treball s'ha centrat en la incorporació d'un apartat d'estadística Bayesiana (*BayesClip)* a l'aplicació *StatClip* amb la intenció d'oferir una eina gratuïta i eficaç per a l'aprenentatge de l'estadística Bayesiana. La recerca elaborada en aquest treball determina la inexistència d'aplicacions semblants a *BayesClip*. Així doncs el present treball ha aconseguit omplir el buit que hi ha en les aplicacions destinades per l'aprenentatge de l'estadística Bayesiana.

Inicialment per tal de dissenyar l'aplicació ha estat necessari l'estudi del temari d'un curs estàndard d'introducció a l'estadística Bayesiana per seleccionar els conceptes més representatius que actualment inclou *BayesClip*. D'altra banda, la implementació de *BayesClip* ha comportat un desenvolupament matemàtic per tal de traduir els elements de la interfície d'usuari al funcionament intern de l'*R.*

Els objectius que es plantejaven a l'inici d'aquest treball han estat assolits amb èxit. Per aconseguir-los s'han adquirit els coneixements necessaris de llenguatge *Shiny* per poder ampliar l'aplicació *StatClip*. La integració de *BayesClip* com a mòdul de *StatClip* ha sigut relativament senzilla gràcies a l'arquitectura interna de *StatClip*, però el fet d'haver-se d'adaptar a la forma de programació aliena a un mateix suposa una dificultat afegida. A més, *BayesClip* ha tingut en compte satisfactòriament les necessitats i requeriments de l'usuari, creant així un disseny lleuger, agradable i intuïtiu. Finalment, les enquestes realitzades a alumnes del grau d'estadística corroboren l'èxit de l'aplicació i la situen com una aplicació útil i innovadora, donada la facilitat de centrar-se en aprendre els conceptes enlloc de centrar-se en la implementació de les tècniques d'estimació.

La implementació de *BayesClip* ha suposat un gran esforç i treball, s'hi ha dedicat la major part de les hores destinades en aquest projecte, i es pot veure reflectit a https://noctilabium.shinyapps.io/BayesClip/.

Personalment al trobar-me amb aquest tema proposat va ser perfecte, ja que no sabia exactament sobre quin tema elaborar el treball de fi de grau. Sempre m'ha agradat molt programar i m'ha resultat relativament senzill. El fet d'haver estudiat *R* durant tot el grau, i ara, poder aprendre les funcions d'una llibreria nova i observar els resultats tant diferents que es poden obtenir amb aquesta m'ha semblat molt motivador. En el grau d'estadística, l'estadística Bayesiana solament s'imparteix en una assignatura i aquesta al principi resulta una mica estranya, però al final em va resultar agradable d'aprendre i vaig obtenir uns bons resultats. Així doncs, poder combinar l'estadística Bayesiana amb programació ha sigut el millor tema que hagués pogut escollir pel meu treball de fi de grau.

9. Bibliografia

[1] RSTUDIO. *Teach yourself Shiny*. [En línia]. 2014. [consulta: 20 de setembre de 2015]. Disponible a: <http://shiny.rstudio.com/tutorial/>.

[2] RSTUDIO. *Gallery*. [En línia]. 2014. [consulta: 22 de setembre de 2015]. Disponible a: <http://shiny.rstudio.com/gallery/>.

[3] RSTUDIO. *Shiny Widgets Gallery*. [En línia]. 2014. [consulta: 22 de setembre de 2015]. Disponible a: <http://shiny.rstudio.com/gallery/widget-gallery.html]>.

[4] RSTUDIO. *Shiny powered dashboards.* [Online]. 2014. [consulta: 24 de setembre de 2015]. Disponible a: <http://rstudio.github.io/shinydashboard/index.html>.

[5] SUBIRANA, I. *Cómo crear aplicaciones con Shiny* (Summer School Facultat de Matemàtiques i Estadística Universitat Politècnica de Catalunya). 2015.

[6] PUIG ORIOL, X. *Apunts assignatura Mètodes Bayesians* 2013.

[7] JOHN WILEY & SONS, LTD. *Bayesian theory*. 2000.

[8] SERRAHIMA DE CAMBRA, E. *Design and development of an app for statistical data analysis learning* (Final Degree Project, directed by Lluis Marco). Barcelona, Universitat Politècnica de Catalunya. 2015.

[9] WAGENMAKERS, EJ. *A First Lesson in Bayesian Inference*. [En línia]. 2015. [consulta: 15 d'octubre de 2015]. Disponible a: <http://87.106.45.173:3838/felix/BayesLessons/BayesianLesson1.Rmd>.

[10] HALTERMAN, A. *Bayes' Rule Calculator*. [En línia]. 2015. [consulta: 15 d'octubre de 2015]. Disponible a: <https://ahalterman.shinyapps.io/BayesCalculator/>.

[11] *Coin Flipping with a Beta Prior.* [En línia]. 2015. [consulta: 15 d'octubre de 2015]. Disponible a: <http://ec2-52-24-93-52.us-west-2.compute.amazonaws.com:3838/bayesian-beta/>.

[12] ERDELY, A. *Bayesian Bernoulli model.* [En línia]. 2015. [consulta: 15 d'octubre de 2015]. Disponible a: <https://aerdely.shinyapps.io/Project/>.

[13] BALSAMIQ STUDIOS LLC. *Balsamiq Mockups*. [En línia]. 2015.[consulta: 21 d'octubre de 2015]. Disponible a: <https://balsamiq.com/>.

[14] RSTUDIO. *File Upload*. [En línia]. 2014. [consulta: 1 de desembre de 2015]. Disponible a: <http://shiny.rstudio.com/gallery/file-upload.html>.

[15] RSTUDIO. *shinyapps.io by RStudio*. [En línia]. 2015. [consulta: 1 de desembre de 2015]. Disponible a: <http://www.shinyapps.io/>.

10. Annex

10.1 UI-body_inference.R

```
#StatClip
#Nuria Busquets, February 2016

#UI-body_inference.R

tab_inference <- tabItem(
  tabName = "inference",
  fluidRow(
    column(width = 4,
           box(
             solidHeader=FALSE,
             status="primary",
             width = NULL,
             selectInput("mod_distr_inference",
                         label="Choose the model's distribution ",
                         choices=list("            "=            1,
"Binomial"="bin","Poisson"="poi",
                                      "Exponential"="exp",
"Normal"="norm"),
                         selected=1)
           ),
           box(
             solidHeader=FALSE,
             status="primary",
             width = NULL,
             # BINOMIAL
             uiOutput("prior_binomial"),
             uiOutput("mean_binomial"),
             uiOutput("sd_binomial"),
             uiOutput("n_binomial"),
             uiOutput("data_binomial"),
             uiOutput("cg"),
             # POISSON
             uiOutput("prior_poisson"),
             uiOutput("mean_poisson"),
             uiOutput("sd_poisson"),
             uiOutput("n_poi"),
             uiOutput("htext_poi_data"),
             uiOutput("data_poi"),
             uiOutput("data_poi2"),
             uiOutput("cg_poi"),
             # NORMAL
             uiOutput("normaldistr"),
             uiOutput("pm"),
             uiOutput("mu_mean"),
             uiOutput("mu_sigma"),
             uiOutput("psig"),
             uiOutput("sig_mean"),
             uiOutput("sig_sigma"),
             uiOutput("htext"),
             uiOutput("selectvar_itp"),
             uiOutput("dat_inference_two_par")
           )
    ),
    column(width = 8,
           tabBox(
             width = NULL,
             id = "tabsetbin",
             tabPanel("Estimation",
                      uiOutput("graf_binomial"),
                      uiOutput("graf2"),
                      uiOutput("plot_mu_sigma"),
                      uiOutput("slow"),
                      uiOutput("post_mu_sigma"),
                      #POISSON
                      uiOutput("dpriori_poisson"),
                      uiOutput("dprioripostvar_poi")
             ),
             tabPanel("Predictive",
```

```
                uiOutput("grafpred1"),
                uiOutput("grafpred2"),
                uiOutput("prior_pred_mu_sigma"),
                uiOutput("post_pred_mu_sigma"),
                #POISSON
                uiOutput("dpredpriori_poi"),
                uiOutput("dpredpost_poi")
        ),
        tabPanel("Summary",
                fluidRow(
                  column(width= 1),
                  column(width= 11,
                      tableOutput("table1"),
                      tableOutput("table_normal_1"),
                      tableOutput("table1_poi"),
                      uiOutput("IC1"),
                      uiOutput("IC1_poi"),
                      column(width=12,
                          tableOutput("table2"),
                          tableOutput("table2_poi")
                      )
                  )
                )
        ),
        tabPanel("Convergence",
                fluidRow(
                  column(width= 1),
                  column(width= 11,
                      uiOutput("convergencia"),
                      tableOutput("Rhat")
                  )
                )
         )
      )
    )
  )
)
```

10.2 UI-body_change_point.R

```
#StatClip
#Nuria Busquets, February 2016

#UI-body_change_point.R

tab_change_point <- tabItem(
tabName = "change-point",
  fluidRow(
    column(width = 4,
           box(
            solidHeader=FALSE,
            status="primary",
            width = NULL,

            selectInput("mod_cp",    label="Choose    the    model's
distribution ",
                        choices=list(" "= 1, "Binomial"="bin"),
                        selected=1)
           ),
           box(
             solidHeader=FALSE,
             status="primary",
             width = NULL,

             uiOutput("priortitle"),
             uiOutput("td1_cp"),
             uiOutput("mean1_cp"),
             uiOutput("sd1_cp"),
             uiOutput("td2_cp"),
             uiOutput("mean2_cp"),
             uiOutput("sd2_cp"),
             uiOutput("rprior_bin"),

             uiOutput("htext_data_ch_bin"),
             uiOutput("data_ch_bin")
             )
```

```
    ),
    column(
      width = 8,
      tabBox(
        width = NULL,
        id = "tabset2popbin",
        tabPanel("Estimation",
                 uiOutput("grafprioris_cp"),
                 uiOutput("grafposterioris_cp")
        ),
        tabPanel("Change Point",
                 uiOutput("rpriorgraf"),
                 uiOutput("slow_cp"),
                 uiOutput("changepointhelp"),
                 uiOutput("grafr_cp")
        ),
        tabPanel("Summary",
                 fluidRow(
                   column(width= 1
                   ),
                   column(width= 11,
                          uiOutput("table1_cp_bin"),
                          uiOutput("IC_cp_bin"),
                      column(width=12,
                          uiOutput("table2_cp_bin"))
                   )
                 )
        ),
        tabPanel("Convergence",
                 uiOutput("convergencet1_cp"),
                 uiOutput("convergencet2_cp"),
                 uiOutput("convergencer_cp")
        )
      )
    )
  )
)
```

10.3 UI-body_compare_two_population.R

```
#StatClip
#Nuria Busquets, February 2016

#UI-body_compare_two_population.R

tab_compare_two_population <- tabItem(
  tabName = "compare-two-population",
  fluidRow(
    column(width = 4,
           box(
             solidHeader=FALSE,
             status="primary",
             width = NULL,

             selectInput("mod_distr_ctp",
                         label="Choose the model's distribution ",
                         choices=list("            "=            1,
"Binomial"="bin","Poisson"="poi",
                                      "Exponential"="exp"),
                         selected=1)
           ),
           box(
             solidHeader=FALSE,
             status="primary",
             width = NULL,
             uiOutput("title_betabin1"),
             uiOutput("td1"),
             uiOutput("mean1"),
             uiOutput("sd1"),
             uiOutput("nt1"),
             uiOutput("td2"),
             uiOutput("mean2"),
             uiOutput("sd2"),
             uiOutput("nt2"),
```

```
        uiOutput("dat"),
        uiOutput("cg_ctp_bin")
      )
),
column(
  width = 8,
  tabBox(
    width = NULL,
    id = "tabset2popbin",
    tabPanel("Estimation",
             uiOutput("graf1twopop1"),
             uiOutput("graf1twopop2")
    ),
    tabPanel("Predictive",
             uiOutput("grafpred1twopopbin"),
             uiOutput("grafpred2twopopbin")
    ),
    tabPanel("Summary",
             fluidRow(
               fluidRow(
                 column(width=1),
                 column(width = 5,
                        h4("Population 1"),
                        uiOutput("table1d1")
                 ),
                 column(width =6,
                        h4("Population 2"),
                        uiOutput("table1d2")
                 )
               ),
               fluidRow(
                 column(width=12,
                        uiOutput("IC1twopopbin")
                 )
               ),
               column(width=1),
               column(width = 5,
                      h4("Population 1"),
                      uiOutput("table2d1")
               ),
               column(width =6 ,
                      h4("Population 2"),
                      uiOutput("table2d2")
               )
             )
    ),
    tabPanel("Difference",
        fluidRow(
             column(width=12,
                 uiOutput("logp"),
                 uiOutput("IC2"),
                 column(width=12,
                     tableOutput("tablediff")
                 )
             )
        )
    )
  )
)
```

10.4 UI-body_load_data_set.R

```
#Statclip
#Eduard Serrahima, May 2015
#UI-body_load_data_set.R
#File defining the data upload tab
#The structure of the page will be the following:
#   Sidebar Menu will be at the left side (as always)
#   The rest of the screen will be divided in a data table (below)
and a
#   top part to define the data upload.
load_data_set <- tabItem(
  tabName = "load",
```

```
  fluidRow(
    #the first fluidRow constains two boxes:
    #   the first one contains the button to paste the data from the
clipboard
    #   and the data conditions the second one contains the sample
data bases,
    #   to choose
    box(
      title=strong(h4("Data upload")),
      status="primary",
      solidHeader=FALSE,
      fileInput('file', 'Choose .csv or .txt file',
                accept=c('text/csv',
                         'text/comma-separated-values,text/plain',
                         '.csv')),
      tags$hr(),
      checkboxInput('header', 'Header', TRUE),
      radioButtons('sep', 'Separator',
                   c(Comma=',',
                     Semicolon=';',
                     Tab='\t'),
                   ';'),
      radioButtons('quote', 'Quote',
                   c(None='',
                     'Double Quote'='"',
                     'Single Quote'="'"),
                   '"')
    ),
    box(
      title=strong(h4("Sample Data")),
      solidHeader=TRUE,
      background="light-blue",
      height=250,
      p("Select a predefined data set to analyze!"),
      actionButton("iris", label="Select Iris data set"),
      br(),
      br(),
      actionButton("mtcars", label="Select Mtcars data set")
    ),
```

```
    fluidRow(
      #the second fluidRow contains a data table with the pasted data
or the
      # selected data set
      box(
        title="Data Table",
        width=12,
        DT::dataTableOutput("load_dataset_table"))
      )
    )
)
```

10.5 UI-body_simple_linear_regression.R

```
#StatClip
#Nuria Busquets, February 2016

#UI-body_simple_linear_regression.R
tab_simple_linear_regression<- tabItem(
  tabName = "simple-linear-regression",
  fluidRow(
    column(width = 4,
      #Plot Conditions box
      box(
        title="Simple linear regression",
        height=1200,
        width = 12,
        solidHeader=FALSE,
        status="primary",
          h3(HTML("y=  &beta;<sub>0</sub>  +    &beta;<sub>1</sub>x
+&epsilon;")),
          br(),
          br(),
          radioButtons("priorknowledge",
```

```
              label = h3("Do you have prior knowledge about the
coefficients?"),
              choices = list("Yes" = 1, "No" = 2),
              selected = 2),
          uiOutput("prior_beta0"),
          uiOutput("prior_beta1"),
          #SelectInput to choose variable y
          uiOutput("select_y_simple_linear_regression"),
          uiOutput("select_x_simple_linear_regression"),
          actionButton("up_slr", label = "Update")
      )
    ),
    column(width = 8,
           tabBox(
             width = NULL,
             id = "tabset_simple_linear_regression",
             tabPanel("Model",
                      tableOutput("m"),
                      uiOutput("title_slr_plot"),
                      uiOutput("postmodel")
             ),
             tabPanel("Plot",
                    uiOutput("postgrafmodel")
             )
           )
    )
  )
)
```

10.6 UI-sidebar.R

```
 #StatClip
#Eduard Serrahima, May 2015

#sidebar.R
#File defining the sidebar for Statclip
#List of menu items
list <- c("Load Data Set","Create Simulated Data","Histogram",
          "Time Series Plot","Dotplot","Pie Chart","Bar
Chart","Scatterplot",
          "Matrix Plot","Boxplot","Bubble Plot","Multi-vari
Chart","Maps",
          "Basic Operations","Probabilities","Correlation",
          "Descriptive Statistics","Goodness of fit","With
Means/Medians",
          "With Variances","With Proportions","Power and Sample
Size",
          "Regression","Inference","Compare Two Population",
          "Simple Linear Regression","Change Point")
sidebar <- dashboardSidebar(

sidebarMenu(
    id = "tabs",
#Data Menu
    menuItem("Data",
             tabName="data",
             icon=icon("table"),
             menuSubItem("Load Data Set",
                         tabName="load",
                         icon=icon("upload")),
             menuSubItem("Create Simulated Data",
                         tabName="simulate",
                         icon=icon("spinner"))
             ),
#Graphs Menu
    menuItem("Graphs",
             tabName="graphs",
             icon=icon("line-chart"),
             menuSubItem("Histogram",
                         tabName="histogram"),
             menuSubItem("Time Series Plot",
                         tabName="timeseries"),
             menuSubItem("Dotplot",
                         tabName="dotplot"),
             menuSubItem("Pie Chart",
                         tabName="piechart"),
             menuSubItem("Bar Chart",
```

```
                    tabName="barchart"),
            menuSubItem("Scatterplot",
                    tabName="scatterplot"),
            menuSubItem("Matrix Plot",
                    tabName="matrixplot"),
            menuSubItem("Boxplot",
                    tabName="boxplot"),
            menuSubItem("Bubble Plot",
                    tabName="bubbleplot"),
            menuSubItem("Multi-vari Chart",
                    tabName="multivari"),
            menuSubItem("Maps",
                    tabName="maps")
            ),
#Comutations Menu
    menuItem("Computations",
            tabName="computations",
            icon=icon("calculator"),
            menuSubItem("Basic Operations",
                    tabName="basicoperations"),
            menuSubItem("Probabilities",
                    tabName="probabilities"),
            menuSubItem("Correlation",
                    tabName="correlation"),
            menuSubItem("Descriptive Statistics",
                    tabName="descriptivestats"),
            menuSubItem("Goodness of fit",
                    tabName="goodnessfit")
            ),
#Statistics Menu
    menuItem("Statistics",
            tabName="statistics",
            icon=icon("bar-chart"),
            menuSubItem("With Means/Medians",
                    tabName="means-medians"),
            menuSubItem("With Variances",
                    tabName="variances"),
            menuSubItem("With Proportions",
                    tabName="proportions"),
            menuSubItem("Power and Sample Size",
                    tabName="power-sample-size"),
            menuSubItem("Regression",
                    tabName="regression")
            ),
#Bayesian Statistics
menuItem("Bayesian Statistics",
         tabName="statistics",
         icon=icon("pie-chart"),
         menuSubItem("Inference",
                    tabName="inference"),
         menuSubItem("Compare Two Population",
                    tabName="compare-two-population "),
         menuSubItem("Simple Linear Regression",
                     tabName="simple-linear-regression"),
         menuSubItem("Change Point",
                     tabName="change-point")
         )
  ),
selectizeInput("searchMenuItem", label="Search Item", choices=list,
               selected=NULL, multiple=FALSE)
)
```

10.7 server-change_point.R

```
#StatClip
#Nuria Busquets, February 2016

#server-change_point.R

output$priortitle <- renderUI({
if(input$mod_cp == "bin"){
  div(
    h4("Prior distributions:"),
    h3(HTML("&theta;<sub>1</sub> ~ Beta(a, b)")),
    h3(HTML("&theta;<sub>2</sub> ~ Beta(&alpha;, &beta;)"))
   )
```

```
}
})

output$td1_cp <- renderUI({
  if(input$mod_cp == "bin"){
    h4("Distribution 1")
  }
})

output$mean1_cp <- renderUI({
  if(input$mod_cp == "bin"){
    sliderInput(inputId="mu1_cp",
                label = "Mean",
                value = 0.5,
                min = 0.01,
                max = 0.99)
  }})

output$sd1_cp <- renderUI({
  if(input$mod_cp == "bin"){
    E <- input$mu1_cp
    maxvalue <- sqrt((-E+2*E^2-E^3)/(E-1))-0.01

    sliderInput(inputId="sd1_cp",
                label="Standard deviation",
                value=sqrt(1/12),
                round=FALSE,
                step=0.001,
                min=0.001,
                max=maxvalue)
  }})

output$td2_cp <- renderUI({
  if(input$mod_cp == "bin"){
    h4("Distribution 2")
  }
})

output$mean2_cp <- renderUI({
  if(input$mod_cp == "bin"){
    sliderInput(inputId="mu2_cp",
                label = "Mean",
                value = 0.5,
                min = 0.01,
                max = 0.99)
  }})

output$sd2_cp <- renderUI({
  if(input$mod_cp == "bin"){
    E <- input$mu2_cp
    maxvalue <- sqrt((-E+2*E^2-E^3)/(E-1))-0.01

    sliderInput(inputId="sd2_cp",
                label="Standard deviation",
                value=sqrt(1/12),
                round=FALSE,
                step=0.001,
                min=0.001,
                max=maxvalue)
  }})

output$rprior_bin <- renderUI({
  if(input$mod_cp == "bin"){
    radioButtons("priorknowledge_r",
                 label = "Do you have prior knowledge about the
change point value?",
                 choices = list("Yes" = 1, "No" = 2),
                 selected = 2)

  }
})

############
####DATA####
############

output$htext_data_ch_bin<- renderPrint({
  if(input$mod_cp == "bin"){
```

```
    div(
    h4("Data"),
      helpText("Previously you have to insert the data in the section
'Data'")
    )
  }
})

output$data_ch_bin <- renderUI({
  if(input$mod_cp == "bin"){
    div(
      if(input$priorknowledge_r == "1"){
          selectInput("r_cp",
                      label="Choose the variable with change point
probabilities",
                      choices=variable_names())
      },
      selectInput("m", label="Choose the variable with parameter N
of binomials",
                  choices=variable_names()),
      selectInput("y_cp", label="Choose the data variable",
                  choices=variable_names()),
      actionButton("up_cp_bin", label = "Update")
    )
  }
})

##############################################
##########  REACTIVE ######################
##############################################

dataInput <- reactive({
  x <- input$up_cp_bin
  if ( x== 0){
    return()
  }
  source("ChangePointFunctions.R", local=TRUE)
  isolate({
      mu1 <- input$mu1_cp
      sd1 <- input$sd1_cp
      mu2 <- input$mu2_cp
      sd2 <- input$sd2_cp
  })

  a <- ((mu1^2)*(1-mu1)/sd1^2)-mu1
  b <- a*(1-mu1)/mu1

  alpha <- ((mu2^2)*(1-mu2)/sd2^2)-mu2
  beta <- alpha*(1-mu2)/mu2

  #DATA
  isolate({
      y <- working_data$data[,match(input$y_cp, variable_names())]
      n <- length(y) #sample size
      #parameter of binomial
      m <- working_data$data[,match(input$m, variable_names())]

      if(input$priorknowledge_r == "1"){
          prob_rp            <-working_data$data[,match(input$r_cp,
variable_names())]
      }else{
          rp <- c(1:n)
          prob_rp <- table(rp)/length(rp)
      }
  })

  #INITIALIZATION
  n.rep <- 10000
  r.support <- 1:n
  r <- rep(NA,n.rep)
  theta1 <- rep(NA,n.rep)
  theta2 <- rep(NA,n.rep)
  theta1[1] <- mu1
  theta2[1] <- mu2
  r[1] <- round(n/2)

  # GIBBS SAMPLING
```

```
  #pb <- winProgressBar(title = "Progress bar", min = 0, max = 100,
width = 300)

  for(t in 2:n.rep){
    theta1[t] <- rbeta(1, sum(y[1:r[t-1]])+a,
                          sum(m[1:r[t-1]])+b-sum(y[1:r[t-1]]))
    theta2[t] <- rbeta(1, sum(y[(r[t-1]+1):n])+alpha ,
                          sum(m[(r[t-1]+1):n])+beta-sum(y[(r[t-
1]+1):n]))
    r[t]   <-   sample(r.support,   size=1,   prob=r.fc(theta1[t],
theta2[t], y))
    #setWinProgressBar(pb, (t/n.rep)*100,
    #                  title=paste( round(t/n.rep*100, 0), "% done"))
  }

  #close(pb)

  #BURN-IN
  To <- 1000
  theta1.bi <- theta1[To:n.rep]
  theta2.bi <- theta2[To:n.rep]
  r.bi <- r[To:n.rep]
  llista    <-    list(a=a,    b=b,    alpha=alpha,    beta=beta,
theta.bi=data.frame(theta1.bi, theta2.bi),
                 r.bi=r.bi, y=y, n=n, m=m)
  return(llista)
})

############
####Tab1###
############

### PRIOR

output$prioris_cp <- renderPlot({
  mu1 <- input$mu1_cp
  sd1 <- input$sd1_cp
  mu2 <- input$mu2_cp
  sd2 <- input$sd2_cp
  a1 <- ((mu1^2)*(1-mu1)/sd1^2)-mu1
  b1 <- a1*(1-mu1)/mu1
  a2 <- ((mu2^2)*(1-mu2)/sd2^2)-mu2
  b2 <- a2*(1-mu2)/mu2
  par(mfrow=c(1,2))
  if(abs(a1-1)<(10^-2) && abs(b1-1)<(10^-2)){
    curve(dbeta(x,1,1),ylab="", xlab = expression(theta[1]),from=0,
to=1,
          lty=1, col="green", lwd = 3)
  }else{
    curve(dbeta(x,a1,b1),ylab="",               xlab               =
expression(theta[1]),from=0, to=1,
          lty=1, col="green", lwd = 3)
  }
  title("Prior distribution 1")
  legend("topright","Beta distribution 1",
         lty = 1,col="green", lwd = 3)
  if(abs(a2-1)<(10^-2) && abs(b2-1)<(10^-2)){
    curve(dbeta(x, 1,1),ylab="", xlab = expression(theta[2]),from=0,
to=1,
          lty=1, col="orange", lwd = 3)
  }else{
    curve(dbeta(x,          a2,b2),ylab="",          xlab          =
expression(theta[2]),from=0, to=1,
          lty=1, col="orange", lwd = 3)
  }
  title("Prior distribution 2")
  legend("topright","Beta distribution 2",
         lty = 1,col="orange", lwd = 3)
  par(mfrow=c(1,1))
})

output$grafprioris_cp <- renderUI({
  if(input$mod_cp == "bin"){
    plotOutput("prioris_cp")}
})

### POSTERIOR
```

```
output$posterioris_cp <- renderPlot({
  if (input$up_cp_bin == 0){
    return()
  }else{
  t1 <- ((dataInput())$theta.bi)[,1]
  t2 <- ((dataInput())$theta.bi)[,2]
  par(mfrow=c(1,2))
  hist(t1,probability=T, col="green", xlab = expression(theta[1]),
       main="Posterior distribution 1")
  hist(t2,probability=T, col="orange", xlab = expression(theta[2]),
       main="Posterior distribution 2")
  par(mfrow=c(1,1))
  }
})

output$grafposterioris_cp <- renderUI({
  if(input$mod_cp == "bin"){
    plotOutput("posterioris_cp")}
})

#########
### R ###
#########

output$rpriorgraf0 <- renderPlot({
  if(input$priorknowledge_r == "1"){
    #yes
      p <- working_data$data[,match(input$r_cp, variable_names())]
      n <- length(p)  #sample size
      rp <- c(1:n)
      x <- table(rp)/length(rp)
      for(i in 1:n){
        x[i] <- p[i]
      }
      prob_rp <- x
  }else{
      #no
      rp <- c(1:10)
      rp <- c(1:6,"7...","n")
      prob_rp <- table(rp)/length(rp)
  }
  barplot(prob_rp ,col="blue",xlab="Chainge point value",
          ylab="Probability",
          main="Prior distribution of change point value",
          ylim=c(0,1),space = 2)
})

output$rpriorgraf <- renderUI({
  if(input$mod_cp == "bin"){
    plotOutput("rpriorgraf0")}
})

output$slow_cp <- renderPrint({
  if(input$mod_cp == "bin"){
    h3("The process is slow, it may take 1 min after press 'update'
button")
  }
 })

output$changepointhelp <- renderPrint({
  if(input$mod_cp == "bin"){
    if (input$up_cp_bin == 0){
      return(h4(" "))
    }else{
     h4("It can be observed that the change point estimation is the
value with higher probability")
  }}
})

output$r_cp <- renderPlot({
  if (input$up_cp_bin == 0){
    return()
  }else{
    r <- ((dataInput())$r.bi)
    p <- table(r)/length(r)
    barplot(p,col="blue",xlab="Chainge point value",
            ylab="Probability",
            main="Posterior distribution of change point value",
```

```
            ylim=c(0,1),space = 2)
   }
})

output$grafr_cp <- renderUI({
  if(input$mod_cp == "bin"){
    plotOutput("r_cp")}
})

#########################################################
####                 ESTIMATION                      ####
#########################################################

output$table1_cp_bin <- renderTable({
  if(input$mod_cp == "bin"){
    if(input$up_cp_bin == 0){
      sortida <- matrix(nrow = 5, ncol = 2)
      colnames(sortida)  <-  c("Prior  distribution  1",  "Prior
distribution 2")
      rownames(sortida) <- c("Alpha", "Beta", "Mean", "Variance",
"Median")
      mu1 <- input$mu1_cp
      sd1 <- input$sd1_cp
      mu2 <- input$mu2_cp
      sd2 <- input$sd2_cp
      a <- ((mu1^2)*(1-mu1)/sd1^2)-mu1
      b <- a*(1-mu1)/mu1
      alpha <- ((mu2^2)*(1-mu2)/sd2^2)-mu2
      beta <- alpha*(1-mu2)/mu2
      sortida[1, 1] <- a
      sortida[2, 1] <- b
      sortida[3, 1] <- mu1
      sortida[4, 1] <- sd1^1
      sortida[5, 1] <- qbeta(0.5, a,b)
      sortida[1, 2] <- alpha
      sortida[2, 2] <- beta
      sortida[3, 2] <- mu2
      sortida[4, 2] <- sd2^1
      sortida[5, 2] <- qbeta(0.5, alpha,beta)
      round(sortida, 3)
    }else{
      mu1 <- input$mu1_cp
      sd1 <- input$sd1_cp
      mu2 <- input$mu2_cp
      sd2 <- input$sd2_cp
      a <- ((dataInput())$a)
      b <- ((dataInput())$b)
      alpha <- ((dataInput())$alpha)
      beta <- ((dataInput())$beta)
      y <- ((dataInput())$y)
      n <- ((dataInput())$n)
      m <- ((dataInput())$m)
      theta1.bi <- ((dataInput())$theta.bi)[,1]
      theta2.bi<- ((dataInput())$theta.bi)[,2]
      r <- ((dataInput())$r.bi)
      mod <- mlv(r,method = "mfv")
      rmod <- as.numeric(mod[1])
      post1 <- c(a+sum(y[1:rmod]), sum(m[1:rmod])-sum(y[1:rmod])+b)
      post2    <-    c(alpha+sum(y[(rmod+1):n]),sum(m[(rmod+1):n])-
sum(y[(rmod+1):n])+beta)
      sortida <- matrix(nrow = 5, ncol = 4)
      colnames(sortida)  <-  c("Prior  distribution  1",  "Posterior
distribution 1",
      "Prior distribution 2","Posterior distribution 2")
      rownames(sortida) <- c("Alpha", "Beta", "Mean", "Variance",
"Median")
      sortida[1, 1] <- a
      sortida[2, 1] <- b
      sortida[3, 1] <- mu1
      sortida[4, 1] <- sd1^1
      sortida[5, 1] <- qbeta(0.5, a,b)
      sortida[1, 2] <- post1[1]
      sortida[2, 2] <- post2[1]
      sortida[3, 2] <- mean(theta1.bi)
      sortida[4, 2] <- sd(theta1.bi)^1
      sortida[5, 2] <- as.numeric(quantile(theta1.bi,0.5))
      sortida[1, 3] <- alpha
      sortida[2, 3] <- beta
```

```
      sortida[3, 3] <- mu2
      sortida[4, 3] <- sd2^1
      sortida[5, 3] <- qbeta(0.5, alpha,beta)
      sortida[1, 4] <- post2[1]
      sortida[2, 4] <- post2[1]
      sortida[3, 4] <- mean(theta2.bi)
      sortida[4, 4] <- sd(theta2.bi)^1
      sortida[5, 4] <- as.numeric(quantile(theta2.bi,0.5))
      return(round(sortida, 3))
    }
  }
})

output$IC_cp_bin <- renderUI({
  if(input$mod_cp == "bin"){
    div(
      column(width=4,
             br(),
             h5("Introduce level of credibility")),
      column(width=2,
             numericInput("icl_cp_bin",label="",value   =   95,min=0,
max=100)),
      column(width=1,
             br(),
             h5("%"))
    )
  }
})

output$table2_cp_bin <- renderTable({
  if(input$mod_cp == "bin"){
    if(input$up_cp_bin == 0){
      l <- input$icl_cp_bin/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      mu1 <- input$mu1_cp
      sd1 <- input$sd1_cp
      mu2 <- input$mu2_cp
      sd2 <- input$sd2_cp
      a <- round(((mu1^2)*(1-mu1)/sd1^2)-mu1,1)
      b <- round(a*(1-mu1)/mu1,1)
      alpha <- round(((mu2^2)*(1-mu2)/sd2^2)-mu2,1)
      beta <- round(alpha*(1-mu2)/mu2,1)
      priori1 <- c(a,b)
      priori2 <- c(alpha,beta)
      s <- matrix(nrow=2,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori1[1], priori1[2]),3)
      s[1,2] <- round(qbeta(alf3, priori1[1], priori1[2]),3)
      s[2,1] <- round(qbeta(alf2, priori2[1], priori2[2]),3)
      s[2,2] <- round(qbeta(alf3, priori2[1], priori2[2]),3)
      rownames(s) <- c("Prior distribution 1","Prior distribution
2")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }else{
      l <- input$icl_cp_bin/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      mu1 <- input$mu1_cp
      sd1 <- input$sd1_cp
      mu2 <- input$mu2_cp
      sd2 <- input$sd2_cp
      a <- ((dataInput())$a)
      b <- ((dataInput())$b)
      alpha <- ((dataInput())$alpha)
      beta <- ((dataInput())$beta)
      priori1 <- c(a,b)
      priori2 <- c(alpha,beta)
      s <- matrix(nrow=4,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori1[1], priori1[2]),3)
      s[1,2] <- round(qbeta(alf3, priori1[1], priori1[2]),3)
      s[2,1] <- round(qbeta(alf2, priori2[1], priori2[2]),3)
      s[2,2] <- round(qbeta(alf3, priori2[1], priori2[2]),3)
      y <- ((dataInput())$y)
      n <- ((dataInput())$n)
      m <- ((dataInput())$m)
```

```
      theta1.bi <- ((dataInput())$theta.bi)[,1]
      theta2.bi<- ((dataInput())$theta.bi)[,2]
      r <- ((dataInput())$r.bi)
      mod <- mlv(r,method = "mfv")
      rmod <- as.numeric(mod[1])
      post1 <- c(a+sum(y[1:rmod]),sum(m[1:rmod])-sum(y[1:rmod])+b)
      post2    <-    c(alpha+sum(y[(rmod+1):n]),sum(m[(rmod+1):n])-
sum(y[(rmod+1):n])+beta)
      s[3,1] <- round(qbeta(alf2, post1[1], post1[2]),3)
      s[3,2] <- round(qbeta(alf3, post1[1], post1[2]),3)
      s[4,1] <- round(qbeta(alf2, post2[1], post2[2]),3)
      s[4,2] <- round(qbeta(alf3, post2[1], post2[2]),3)
      rownames(s) <- c("Prior distribution 1","Prior distribution
2",
                       "Posterior   distribution   1",   "Posterior
distribution 2")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }
  }
})

#########################################################
####                  CONVERGENCE                    ####
#########################################################

output$convergencet1_cp0 <- renderPlot({
  if (input$up_cp_bin == 0){
    return()
  }else{
    t1 <- ((dataInput())$theta.bi)[,1]
    plot(t1, type="l", ylab="Iteration", xlab=expression(theta[1]),
col="green")
    abline(h=mean(t1), col="black")
  }
})

output$convergencet1_cp <- renderUI({
  if(input$mod_cp == "bin"){
    plotOutput("convergencet1_cp0")}
})

output$convergencet2_cp0 <- renderPlot({
  if (input$up_cp_bin == 0){
    return()
  }else{
    t2 <- ((dataInput())$theta.bi)[,2]
    plot(t2,                type="l",                ylab="Iteration",
xlab=expression(theta[2]),col="orange")
    abline(h=mean(t2), col="black")
  }
})

output$convergencet2_cp <- renderUI({
  if(input$mod_cp == "bin"){
    plotOutput("convergencet2_cp0")}
})


output$convergencer_cp0 <- renderPlot({
  if (input$up_cp_bin == 0){
    return()
  }else{
    r <- ((dataInput())$r.bi)
    plot(r,   type="l",   ylab="Iteration",   xlab="Change   point",
col="blue")
    abline(h=mean(r), col="black")
  }
})

output$convergencer_cp <- renderUI({
  if(input$mod_cp == "bin"){
    plotOutput("convergencer_cp0")}
})
```

10.8 server-compare_two_population.R

```
#StatClip
#Nuria Busquets, February 2016

#server-compare_two_population.R
output$title_betabin1 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    div(
      h3("Prior distribution:"),
      h3(HTML("&theta; ~ Beta(&alpha;, &beta;)")),
      h3("Priror predictive distribution:"),
      h3(HTML("P   <sub>&pi;</sub>   (y)   ~   BetaBinomial(&alpha;,
&beta;)"))
    )
  }
})

output$td1 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    h4("Population 1")
  }
})

output$mean1 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    sliderInput(inputId="mu1",
                label = "Mean",
                value = 0.5,
                min = 0.01,
                max = 0.99)
  }})

output$sd1 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    E <- input$mu1
    maxvalue <- sqrt((-E+2*E^2-E^3)/(E-1))-0.01
    sliderInput(inputId="sd1",
                label="Standard deviation",
                value=sqrt(1/12),
                round=FALSE,
                step=0.001,
                min=0.001,
                max=maxvalue)
  }})

output$nt1 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    sliderInput(inputId="nt1",
                label="N number of trials (zero or more)",
                value=10,
                min=0,
                max=100)
  }})

output$td2 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    h4("Population 2")
  }
})

output$mean2 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    sliderInput(inputId="mu2",
                label = "Mean",
                value = 0.5,
                min = 0.01,
                max = 0.99)
  }
})

output$sd2 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    E <- input$mu2
    maxvalue <- sqrt((-E+2*E^2-E^3)/(E-1))-0.01
```

```
    sliderInput(inputId="sd2",
                label="Standard deviation",
                value=sqrt(1/12),
                round=FALSE,
                step=0.001,
                min=0.001,
                max=maxvalue)
  }})

output$nt2 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    sliderInput(inputId="nt2",
                label="N number of trials (zero or more)",
                value=10,
                min=0,
                max=100)
  }})

output$dat <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    div(
      h4("Data"),
      column(width =3,
             numericInput("nsample1",   label   =   "N   1",   value   =
100,min=1, max=1000)
      ),
      column(width=3,
             numericInput("y1",label = "Y 1", value = 9,min=0)
      ),
      column(width =3,
             numericInput("nsample2",   label   =   "N   2",   value   =
100,min=1, max=1000)
      ),
      column(width=3,
             numericInput("y2",label = "Y 2", value = 1,min=0)
      ),
      fluidRow(
             column(width=4),
             column(width=8,
                    actionButton("up_ctp_bin", label = "Update")
             )
      )
    )
  }
})

############
####Tab1####
############

output$dp <- renderPlot({
  a1 <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
  b1 <- a1*(1-input$mu1)/input$mu1
  a2 <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
  b2 <- a2*(1-input$mu2)/input$mu2
  par(mfrow=c(1,2))
  if(abs(a1-1)<(10^-2) && abs(b1-1)<(10^-2)){
    curve(dbeta(x,1,1),ylab="", xlab = expression(theta[1]),from=0,
to=1,
          lty=1, col="green", lwd = 3)
  }else{
  curve(dbeta(x,a1,b1),ylab="", xlab = expression(theta[1]),from=0,
to=1,
        lty=1, col="green", lwd = 3)
  }
  title("Prior distribution 1")
  legend("topright","Beta distribution 1",
         lty = 1,col="green", lwd = 3)
  if(abs(a2-1)<(10^-2) && abs(b2-1)<(10^-2)){
    curve(dbeta(x, 1,1),ylab="", xlab = expression(theta[2]),from=0,
to=1,
          lty=1, col="orange", lwd = 3)
  }else{
    curve(dbeta(x,          a2,b2),ylab="",          xlab          =
expression(theta[2]),from=0, to=1,
          lty=1, col="orange", lwd = 3)
  }
  title("Prior distribution 2")
```

```
  legend("topright","Beta distribution 2",
         lty = 1,col="orange", lwd = 3)
  par(mfrow=c(1,1))
})

output$graf1twopop1 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    plotOutput("dp")}
})

output$cg_ctp_bin <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    if (input$up_ctp_bin == 0){
      return()
    }
    isolate(
      checkboxGroupInput("checkGroup_ctp_bin", label = h4
                         ("Posterior distribution's graphic"),
                         choices = list("Prior" = 1, "Likelihood" =
2),
                         selected = c(1,2)
      )
    )
  }
})

output$dprioripostvarctpbin <- renderPlot({
  if(input$mod_distr_ctp == "bin"){
    if (input$up_ctp_bin == 0){
      return()
    }
  }
  isolate({
    suport <- seq(from = 0, to = 1, 0.001)
    y1 <- input$y1
    y2 <- input$y2
    n1 <- input$nsample1
    n2 <- input$nsample2
    mu1 <- input$mu1
    va1 <- input$sd1^2
    mu2 <- input$mu2
    va2 <- input$sd2^2
    vers1 <- dbinom(y1,n1,prob=suport)
    vers2 <- dbinom(y2,n2,prob=suport)
    K1 <- integrate(function(x)dbinom(y1,n1,x), lower = 0,
                    upper = 1, subdivisions=1000)$value
    K2 <- integrate(function(x)dbinom(y2,n2,x), lower = 0,
                    upper = 1, subdivisions=1000)$value
    vers.resc1 <- vers1/K1
    vers.resc2 <- vers2/K2
    ap1 <- ((mu1^2)*(1-mu1)/va1)-mu1
    (bp1 <- ap1*(1-mu1)/mu1)
    (ap2 <- ((mu2^2)*(1-mu2)/va2)-mu2)
    (bp2 <- ap2*(1-mu2)/mu2)
    dist.priori1  <- dbeta(suport, round(ap1,1),round(bp1,1))
    dist.priori2  <- dbeta(suport, round(ap2,1),round(bp2,1))
    posteriori1 <- c(ap1 + y1, bp1 + (n1-y1))
    dist.posteriori1<-dbeta(suport,posteriori1[1],posteriori1[2])
    posteriori2 <- c(ap2 + y2, bp2 + (n2-y2))
    dist.posteriori2<-dbeta(suport,posteriori2[1],posteriori2[2])
  })
  par(mfrow=c(1,2))
  plot(suport,dist.posteriori1,                                ylab="",
xlab=expression(theta[1]),type="l",
       ylim=c(0,     max(dist.posteriori1,vers.resc1)),     lty=1,
col="red", lwd = 3)
  if(sum(as.numeric(input$checkGroup_ctp_bin))==1){
    lines(suport, dist.priori1, lty=1, col="green", lwd = 3)}
  if(sum(as.numeric(input$checkGroup_ctp_bin))==2){
    lines(suport, vers.resc1, lty=3, col="black", lwd = 2)}
  if(sum(as.numeric(input$checkGroup_ctp_bin))==3){
    lines(suport, dist.priori1,lty=1, col="green", lwd = 3)
    lines(suport, vers.resc1, lty=3, col="black", lwd = 2)
  }
  title("Posterior distribution 1 ")
  legend("topright", c("posterior","prior","likelihood"),
         lty   =   c(1,1,3),   col=c("red","green","black"),   lwd=
c(3,3,2))
```

```
  plot(suport,dist.posteriori2,                                ylab="",
xlab=expression(theta[2]),type="l",
       ylim=c(0,      max(dist.posteriori2,vers.resc2)),      lty=1,
col="red", lwd = 3)
  if(sum(as.numeric(input$checkGroup_ctp_bin))==1){
    lines(suport, dist.priori2, lty=1, col="orange", lwd = 3)}
  if(sum(as.numeric(input$checkGroup_ctp_bin))==2){
    lines(suport, vers.resc2, lty=3, col="black", lwd = 2)}
  if(sum(as.numeric(input$checkGroup_ctp_bin))==3){
    lines(suport, dist.priori2,lty=1, col="orange", lwd = 3)
    lines(suport, vers.resc2, lty=3, col="black", lwd = 2)
  }
  title("Posterior distribution 2 ")
  legend("topright", c("posterior","prior","likelihood"),
         lty   =   c(1,1,3),   col=c("red","orange","black"),   lwd=
c(3,3,2))
  par(mfrow=c(1,1))
})

output$graf1twopop2 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    plotOutput("dprioripostvarctpbin")}
})

#################
####Predictive###
#################

output$dpredpriori2ctpbin <- renderPlot({
  a1 <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
  b1 <- a1*(1-input$mu1)/input$mu1
  a2 <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
  b2 <- a2*(1-input$mu2)/input$mu2
  v1 <- dbetabinom.ab(1:1000,size=input$nt1,a1,b1)
  v2 <- dbetabinom.ab(1:1000,size=input$nt2,a2,b2)
  par(mfrow=c(1,2))
  plot( v1,ty="h", col="green",xlim=c(0,input$nt1),xlab="",ylab="")
  title("Prior predictive distribution 1 \n BetaBinomial")
  plot(                                                        v2,ty="h",
col="orange",xlim=c(0,input$nt2),xlab="",ylab="")
  title("Prior predictive distribution 2 \n BetaBinomial")
  par(mfrow=c(1,1))
})

output$grafpred1twopopbin <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    plotOutput("dpredpriori2ctpbin")}
})

output$dpredpostctpbin <- renderPlot({
  if(input$mod_distr_ctp == "bin"){
    if (input$up_ctp_bin == 0){
      return()
    }
  }
  isolate({
    a1 <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
    b1 <- a1*(1-input$mu1)/input$mu1
    a2 <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
    b2 <- a2*(1-input$mu2)/input$mu2
    posteriori1 <- c(a1 + input$y1,b1 + (input$nsample1-input$y1))
    posteriori2 <- c(a2 + input$y2, b2 + (input$nsample2-input$y2))
    v1                                                          <-
dbetabinom.ab(1:1000,size=input$nsample1,posteriori1[1],posteriori1
[2])
    v2                                                          <-
dbetabinom.ab(1:1000,size=input$nsample2,posteriori2[1],posteriori2
[2])
    par(mfrow=c(1,2))
    plot(v1,ty="h",
col="red",xlim=c(0,input$nsample1),xlab="",ylab="")
    title("Posterior predictive distribution 1 \n BetaBinomial")
    plot(v2,ty="h",
col="red",,xlim=c(0,input$nsample2),xlab="",ylab="")
    title("Posterior predictive distribution 2 \n BetaBinomial")
    par(mfrow=c(1,1))
  })
```

```
})

output$grafpred2twopopbin <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    plotOutput("dpredpostctpbin")}
})

#########################################################
####                 ESTIMATION                      ####
#########################################################

output$table1d1 <- renderTable({
  if(input$mod_distr_ctp == "bin"){
    if(input$up_ctp_bin == 0){
      sortida1 <- matrix(nrow = 5, ncol = 1)
      colnames(sortida1) <- c("Prior")
      rownames(sortida1)  <-  c("Alpha","Beta",  "Mean",  "Variance",
"Median")
      a1 <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
      b1 <- a1*(1-input$mu1)/input$mu1
      priori1 <- c(a1,b1)
      sortida1[1, 1] <- priori1[1]
      sortida1[2, 1] <- priori1[2]
      sortida1[3, 1] <- priori1[1]/(priori1[1] + priori1[2])
      sortida1[4,                          1]                          <-
(priori1[1]*priori1[2])/(((priori1[1]+priori1[2])^2)*(priori1[1]+pr
iori1[2]+1))
      sortida1[5, 1] <- qbeta(0.5, priori1[1], priori1[2])
      round(sortida1, 3)
    }else{
      a1 <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
      b1 <- a1*(1-input$mu1)/input$mu1
      priori1 <- c(a1,b1)
      posteriori1  <-  c(a1  +  input$y1,  b1  +  (input$nsample1-
input$y1))
      sortida1 <- matrix(nrow = 5, ncol = 2)
      sortida1[1, 1] <- priori1[1]
      sortida1[2, 1] <- priori1[2]
      sortida1[3, 1] <- priori1[1]/(priori1[1] + priori1[2])
      sortida1[4,                          1]                          <-
(priori1[1]*priori1[2])/(((priori1[1]+priori1[2])^2)*(priori1[1]+pr
iori1[2]+1))
      sortida1[5, 1] <- qbeta(0.5, priori1[1], priori1[2])
      sortida1[1, 2] <- posteriori1[1]
      sortida1[2, 2] <- posteriori1[2]
      sortida1[3,    2]    <-    posteriori1[1]/(posteriori1[1]    +
posteriori1[2])
      sortida1[4,                          2]                          <-
(posteriori1[1]*posteriori1[2])/(((posteriori1[1]+posteriori1[2])^2
)*(posteriori1[1]+posteriori1[2]+1))
      sortida1[5, 2] <- qbeta(0.5, posteriori1[1], posteriori1[2])
      colnames(sortida1) <- c("Prior","Posterior")
      rownames(sortida1)  <-  c("Alpha","Beta",  "Mean",  "Variance",
"Median")
      return(round(sortida1, 3))
    }
  }
})

output$table1d2 <- renderTable({
  if(input$mod_distr_ctp == "bin"){
    if(input$up_ctp_bin == 0){
      sortida2 <- matrix(nrow = 5, ncol = 1)
      colnames(sortida2) <- c("Prior")
      rownames(sortida2)  <-  c("Alpha","Beta",  "Mean",  "Variance",
"Median")
      a2 <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
      b2 <- a2*(1-input$mu2)/input$mu2
      priori2 <- c(a2,b2)
      sortida2[1, 1] <- priori2[1]
      sortida2[2, 1] <- priori2[2]
      sortida2[3, 1] <- priori2[1]/(priori2[1] + priori2[2])
      sortida2[4,                          1]                          <-
(priori2[1]*priori2[2])/(((priori2[1]+priori2[2])^2)*(priori2[1]+pr
iori2[2]+1))
      sortida2[5, 1] <- qbeta(0.5, priori2[1], priori2[2])
      round(sortida2, 3)
    }else{
```

```
      a2 <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
      b2 <- a2*(1-input$mu2)/input$mu2
      priori2 <- c(a2,b2)
      posteriori2  <-  c(a2  +  input$y2,  b2  +  (input$nsample2-
input$y2))
      sortida2 <- matrix(nrow = 5, ncol = 2)
      sortida2[1, 1] <- priori2[1]
      sortida2[2, 1] <- priori2[2]
      sortida2[3, 1] <- priori2[1]/(priori2[1] + priori2[2])
      sortida2[4,                          1]                          <-
(priori2[1]*priori2[2])/(((priori2[1]+priori2[2])^2)*(priori2[1]+pr
iori2[2]+1))
      sortida2[5, 1] <- qbeta(0.5, priori2[1], priori2[2])
      sortida2[1, 2] <- posteriori2[1]
      sortida2[2, 2] <- posteriori2[2]
      sortida2[3,    2]    <-    posteriori2[1]/(posteriori2[1]    +
posteriori2[2])
      sortida2[4,                          2]                          <-
(posteriori2[1]*posteriori2[2])/(((posteriori2[1]+posteriori2[2])^2
)*(posteriori2[1]+posteriori2[2]+1))
      sortida2[5, 2] <- qbeta(0.5, posteriori2[1], posteriori2[2])
      colnames(sortida2) <- c("Prior","Posterior")
      rownames(sortida2)  <-  c("Alpha","Beta",  "Mean",  "Variance",
"Median")
      return(round(sortida2, 3))
    }
  }
})

output$IC1twopopbin <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    div(
      column(width=4,
             br(),
             h5("Introduce level of credibility")),
      column(width=2,
             numericInput("icl1twopopbin",label="",value          =
95,min=0, max=100)),
      column(width=1,
             br(),
             h5("%"))
    )
  }
})

output$table2d1 <- renderTable({
  if(input$mod_distr_ctp == "bin"){
    if(input$up_ctp_bin == 0){
      l <- input$icl1twopopbin/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
      b <- a*(1-input$mu1)/input$mu1
      priori <- c(a,b)
      s <- matrix(nrow=2,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qbeta(alf3, priori[1], priori[2]),3)
      v <- rbetabinom.ab(1:1000,size=input$nt1,priori[1],priori[2])
      s[2,1] <- as.numeric(quantile(v,alf2))
      s[2,2] <- as.numeric(quantile(v,alf3))
      rownames(s) <-c("Prior","Prior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }else{
      l <- input$icl1twopopbin/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
      b <- a*(1-input$mu1)/input$mu1
      ap  <-  a + input$y1
      bp  <-  b + (input$nsample1-input$y1)
      priori <- c(a,b)
      posteriori <- c(ap,bp)
      s <- matrix(nrow=4,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qbeta(alf3, priori[1], priori[2]),3)
```

```
      v <- rbetabinom.ab(1:1000,size=input$nt1,priori[1],priori[2])
      s[2,1] <- as.numeric(quantile(v,alf2))
      s[2,2] <- as.numeric(quantile(v,alf3))
      s[3,1] <- round(qbeta(alf2, posteriori[1], posteriori[2]),3)
      s[3,2] <- round(qbeta(alf3, posteriori[1], posteriori[2]),3)
      v2                                                          <-
rbetabinom.ab(1:1000,size=input$nsample1,posteriori[1],posteriori[2
])
      s[4,1] <- as.numeric(quantile(v2,alf2))
      s[4,2] <- as.numeric(quantile(v2,alf3))
      rownames(s) <- c("Prior","Prior predictive","Posterior",
                       "Posterior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }
  }
})

output$table2d2 <- renderTable({
  if(input$mod_distr_ctp == "bin"){
    if(input$up_ctp_bin == 0){
      l <- input$icl1twopopbin/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
      b <- a*(1-input$mu2)/input$mu2
      priori <- c(a,b)
      s <- matrix(nrow=2,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qbeta(alf3, priori[1], priori[2]),3)
      v <- rbetabinom.ab(1:1000,size=input$nt2,priori[1],priori[2])
      s[2,1] <- as.numeric(quantile(v,alf2))
      s[2,2] <- as.numeric(quantile(v,alf3))
      rownames(s) <- c("Prior","Prior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }else{
      l <- input$icl1twopopbin/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
      b <- a*(1-input$mu2)/input$mu2
      ap  <-  a + input$y2
      bp  <-  b + (input$nsample2-input$y2)
      priori <- c(a,b)
      posteriori <- c(ap,bp)
      s <- matrix(nrow=4,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qbeta(alf3, priori[1], priori[2]),3)
      v <- rbetabinom.ab(1:1000,size=input$nt2,priori[1],priori[2])
      s[2,1] <- as.numeric(quantile(v,alf2))
      s[2,2] <- as.numeric(quantile(v,alf3))
      s[3,1] <- round(qbeta(alf2, posteriori[1], posteriori[2]),3)
      s[3,2] <- round(qbeta(alf3, posteriori[1], posteriori[2]),3)
      v2                                                          <-
rbetabinom.ab(1:1000,size=input$nsample2,posteriori[1],posteriori[2
])
      s[4,1] <- as.numeric(quantile(v2,alf2))
      s[4,2] <- as.numeric(quantile(v2,alf3))
      rownames(s) <- c("Prior","Prior predictive","Posterior",
                       "Posterior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }
  }
})

#######################################################
####                DIFFERENCE                     ####
#######################################################

output$log <- renderPlot({
  if(input$mod_distr_ctp == "bin"){
    if (input$up_ctp_bin == 0){
      return()
    }
```

```
    else{
      a1 <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
      b1 <- a1*(1-input$mu1)/input$mu1
      a2 <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
      b2 <- a2*(1-input$mu2)/input$mu2
      priori1 <-c(a1 , b1)
      priori2 <-c(a2 , b2)
      posteriori1                                                    <-
c(priori1[1]+input$y1,priori1[2]+(input$nsample1-input$y1))
      posteriori2                                                    <-
c(priori2[1]+input$y2,priori2[2]+(input$nsample2-input$y2))
      n.sim <- 100000
      z1 <- rbeta(n.sim, posteriori1[1], posteriori1[2])
      z2 <- rbeta(n.sim, posteriori2[1], posteriori2[2])
      l <- log(z1/z2)
      plot(density(l, adjust=1.2),ylab="",xlab="",
           main="Distribution of the logarithm of the ratio of
distributions",
           col="blue")
      abline(v=0, col="red")
    }
  }
})

output$logp <- renderUI({
  plotOutput("log")
})

output$IC2 <- renderUI({
  if(input$mod_distr_ctp == "bin"){
    if(input$up_ctp_bin == 0){
    }else{
      div(
        column(width=4,
               br(),
               h5("Introduce level of credibility")),
        column(width=2,
               numericInput("icl2",label="",value    =    95,min=0,
max=100)),
        column(width=1,
               br(),
               h5("%"))
      )
    }
  }
})

output$tablediff <- renderTable({
  if(input$mod_distr_ctp == "bin"){
    if (input$up_ctp_bin == 0){
      return()
    }
    else{
      l <- input$icl2/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a1 <- ((input$mu1^2)*(1-input$mu1)/input$sd1^2)-input$mu1
      b1 <- a1*(1-input$mu1)/input$mu1
      a2 <- ((input$mu2^2)*(1-input$mu2)/input$sd2^2)-input$mu2
      b2 <- a2*(1-input$mu2)/input$mu2
      priori1 <-c(a1 , b1)
      priori2 <-c(a2 , b2)
      posteriori1                                                    <-
c(priori1[1]+input$y1,priori1[2]+(input$nsample1-input$y1))
      posteriori2                                                    <-
c(priori2[1]+input$y2,priori2[2]+(input$nsample2-input$y2))
      n.sim <- 100000
      z1 <- rbeta(n.sim, posteriori1[1], posteriori1[2])
      z2 <- rbeta(n.sim, posteriori2[1], posteriori2[2])
      lo <- log(z1/z2)
      loo <- sort(lo)
      c <- c(quantile(loo,alf2), quantile(loo,alf3))
      s <- matrix(nrow=1,ncol=2)
      s[1,1] <- round(as.numeric(c[1]),3)
      s[1,2] <- round(as.numeric(c[2]),3)
      rownames(s) <- c("Log of the differences")
      colnames(s) <- c("Lower bound", "Upper bound")
```

```
      return(s)
    }
  }
})
```

10.9 server-inference.R

```
#StatClip
#Nuria Busquets, February 2016

#server-inference.R
source("Server_files/server-inference_poisson.R", local=TRUE)
source("Server_files/server-inference_binomial.R", local=TRUE)
source("Server_files/server-inference_normal.R", local=TRUE)
```

10.10 server-inference_binomial.R

```
#StatClip
#Nuria Busquets, February 2016

#server-inference_binomial.R

output$prior_binomial <- renderUI({
  if(input$mod_distr_inference == "bin"){
    div(
      h3("Prior distribution:"),
      h3(HTML("&theta; ~ Beta(&alpha;, &beta;)")),
      h3("Priror predictive distribution:"),
      h3(HTML("P   <sub>&pi;</sub>   (y)   ~   BetaBinomial(&alpha;,
&beta;)"))
    )
  }
})

output$mean_binomial <- renderUI({
  if(input$mod_distr_inference == "bin"){
    sliderInput(inputId="mu",
                label = "Mean",
                value = 0.5,
                min = 0.01,
                max = 0.99)
  }
})

output$sd_binomial <- renderUI({
  if(input$mod_distr_inference == "bin"){
    E <- input$mu
    maxvalue <- sqrt((-E+2*E^2-E^3)/(E-1))-0.01
    sliderInput(inputId="sd",
                label="Standard deviation",
                value=sqrt(1/12),
                round=FALSE,
                step=0.001,
                min=0.001,
                max=maxvalue)
  }})

output$n_binomial <- renderUI({
  if(input$mod_distr_inference == "bin"){
    sliderInput(inputId="nn",
                label="N number of trials (zero or more)",
                value=10,
                min=0,
                max=100)
  }})

############
### DATA ###
############
output$data_binomial <- renderUI({
  if(input$mod_distr_inference == "bin"){
    div(
      h4("Data"),
      column(width =4,
```

```
            numericInput("nsample", label = "N", value = 100,min=1,
max=1000)
      ),
      column(width=4,
             numericInput("ymostra",label = "Y", value = 9,min=0)
      ),
      column(width=4,
             br(),
             actionButton("up_inferential", label = "Update")
      )
    )
  }
})

output$dpriori_binomial <- renderPlot({
  if(input$mod_distr_inference == "bin"){
    a <- (((input$mu^2)*(1-input$mu)/input$sd^2)-input$mu)
    b <- ((a*(1-input$mu)/input$mu))
    if(abs(a-1)<(10^-2) && abs(b-1)<(10^-2)){
      curve(dbeta(x, 1,1),from=0, to=1,lty=1, col="blue", lwd = 3,
ylab="", xlab = expression(theta))
    }else{
      curve(dbeta(x, a,b),from=0, to=1,lty=1, col="blue", lwd = 3,
ylab="", xlab = expression(theta))
    }
    title("Prior distribution ")
    legend("topright", "Beta distribution", lty = 1,col="blue", lwd
= 3)
  }
})

output$graf_binomial <- renderUI({
  if(input$mod_distr_inference == "bin"){
    plotOutput("dpriori_binomial")}
})

output$cg <- renderUI({
  if(input$mod_distr_inference == "bin"){
    if (input$up_inferential == 0){
      return()
    }
    isolate(
      checkboxGroupInput("checkGroup",    label    =    h3("Posterior
distribution's graphic"),
                         choices = list("Prior" = 1, "Likelihood" =
2),
                         selected = c(1,2)
      )
    )
  }
})

output$dprioripostvar <- renderPlot({
  if(input$mod_distr_inference == "bin"){
    if (input$up_inferential == 0){
      return()
    }
  }
  isolate({
    suport <- seq(from = 0, to = 1, 0.001)
    y <- input$ymostra
    n <- input$nsample
    mu <- input$mu
    va <- (input$sd)^2
    vers <- dbinom(y,n,prob=suport)
    K <- integrate(function(x)dbinom(y,n,x), lower = 0, upper = 1,
                   subdivisions=1000)$value
    vers.resc <- vers/K
    ap <- round(((mu^2)*(1-mu)/va)-mu,1)
    bp <- round(ap*(1-mu)/mu,1)
    dist.priori  <- dbeta(suport, ap,bp)
    posteriori <- c(ap + y, bp + (n-y))
    dist.posteriori<-dbeta(suport,posteriori[1],posteriori[2])
  })
  plot(suport,dist.posteriori,                                ylab="",
xlab=expression(theta),type="l",
       ylim=c(0, max(dist.posteriori,vers.resc)), lty=1, col="red",
lwd = 3)
```

```
  if(sum(as.numeric(input$checkGroup))==1){
    lines(suport, dist.priori, lty=1, col="blue", lwd = 3)}
  if(sum(as.numeric(input$checkGroup))==2){
    lines(suport, vers.resc, lty=3, col="black", lwd = 2)}
  if(sum(as.numeric(input$checkGroup))==3){
    lines(suport, dist.priori,lty=1, col="blue", lwd = 3)
    lines(suport, vers.resc, lty=3, col="black", lwd = 2)
  }
  title("Posterior distribution ")
  legend("topright", c("posterior","prior","likelihood"),
         lty = c(1,1,3), col=c("red","blue","black"), lwd= c(3,3,2))
})

output$graf2 <- renderUI({
  if(input$mod_distr_inference == "bin"){
    plotOutput("dprioripostvar")}
})

output$dpredpriori <- renderPlot({
  a <- ((input$mu^2)*(1-input$mu)/input$sd^2)-input$mu
  b <- a*(1-input$mu)/input$mu
  v <- dbetabinom.ab(1:1000,size=input$nn,a,b)
  plot(v,ty="h", col="blue",xlim=c(0,input$nn),xlab="",ylab="")
  title("Prior predictive distribution \n BetaBinomial")
})

output$grafpred1 <- renderUI({
  if(input$mod_distr_inference == "bin"){
    plotOutput("dpredpriori")}
})


####### POSTERIOR PREDICTIVE #######
output$dpredpost_bin <- renderPlot({
  if(input$mod_distr_inference == "bin"){
    if (input$up_inferential == 0){
      return()
    }
  }
  isolate({
    a <- ((input$mu^2)*(1-input$mu)/input$sd^2)-input$mu
    b <- a*(1-input$mu)/input$mu
    posteriori <- c(a + input$ymostra, b + (input$nsample-
input$ymostra))
    v <-
dbetabinom.ab(1:1000,size=input$nsample,posteriori[1],posteriori[2]
)
    plot(v,ty="h",
col="blue",xlim=c(0,input$nsample),xlab="",ylab="")
    title("Posterior predictive distribution \n BetaBinomial")
  })
})

output$grafpred2 <- renderUI({
  if(input$mod_distr_inference == "bin"){
    plotOutput("dpredpost_bin")}
})

output$table1 <- renderTable({
  if(input$mod_distr_inference == "bin"){
    if(input$up_inferential == 0){
      sortida <- matrix(nrow = 5, ncol = 1)
      colnames(sortida) <- c("Prior")
      rownames(sortida) <- c("Alpha","Beta", "Mean", "Variance",
"Median")
      a <- ((input$mu^2)*(1-input$mu)/input$sd^2)-input$mu
      b <- a*(1-input$mu)/input$mu
      priori <- c(a,b)
      sortida[1, 1] <- priori[1]
      sortida[2, 1] <- priori[2]
      sortida[3, 1] <- priori[1]/(priori[1] + priori[2])
      sortida[4, 1] <-
(priori[1]*priori[2])/(((priori[1]+priori[2])^2)*(priori[1]+priori[
2]+1))
      sortida[5, 1] <- qbeta(0.5, priori[1], priori[2])
      return(sortida)
    }else{
      a <- ((input$mu^2)*(1-input$mu)/input$sd^2)-input$mu
```

```
      b <- a*(1-input$mu)/input$mu
      priori <- c(a,b)
      posteriori  <-  c(a  +  input$ymostra,  b  +  (input$nsample-
input$ymostra))
      sortida <- matrix(nrow = 5, ncol = 2)
      sortida[1, 1] <- priori[1]
      sortida[2, 1] <- priori[2]
      sortida[3, 1] <- priori[1]/(priori[1] + priori[2])
      sortida[4,                          1]                          <-
(priori[1]*priori[2])/(((priori[1]+priori[2])^2)*(priori[1]+priori[
2]+1))
      sortida[5, 1] <- qbeta(0.5, priori[1], priori[2])
      sortida[1, 2] <- posteriori[1]
      sortida[2, 2] <- posteriori[2]
      sortida[3, 2] <- posteriori[1]/(posteriori[1] + posteriori[2])
      sortida[4,                          2]                          <-
(posteriori[1]*posteriori[2])/(((posteriori[1]+posteriori[2])^2)*(p
osteriori[1]+posteriori[2]+1))
      sortida[5, 2] <- qbeta(0.5, posteriori[1], posteriori[2])
      colnames(sortida) <- c("Prior","Posterior")
      rownames(sortida)  <-  c("Alpha","Beta",  "Mean",  "Variance",
"Median")
      return(round(sortida, 3))
    }
  }
})

output$IC1 <- renderUI({
  if(input$mod_distr_inference == "bin"){
    div(
      column(width=4,
             br(),
             h5("Introduce level of credibility")),
      column(width=2,
             numericInput("icl1",label="",value     =     95,min=0,
max=100)),
      column(width=1,
             br(),
             h5("%"))
    )
  }
})

output$table2 <- renderTable({
  if(input$mod_distr_inference == "bin"){
    if(input$up_inferential == 0){
      l <- input$icl1/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- ((input$mu^2)*(1-input$mu)/input$sd^2)-input$mu
      b <- a*(1-input$mu)/input$mu
      priori <- c(a,b)
      s <- matrix(nrow=2,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qbeta(alf3, priori[1], priori[2]),3)
      v <- rbetabinom.ab(1:1000,size=input$nn,priori[1],priori[2])
      s[2,1] <- as.numeric(quantile(v,alf2))
      s[2,2] <- as.numeric(quantile(v,alf3))
      rownames(s) <- c("Prior","Prior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }else{
      l <- input$icl1/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- ((input$mu^2)*(1-input$mu)/input$sd^2)-input$mu
      b <- a*(1-input$mu)/input$mu
      ap  <-  a + input$ymostra
      bp  <-  b + (input$nsample-input$ymostra)
      priori <- c(a,b)
      posteriori <- c(ap,bp)
      s <- matrix(nrow=4,ncol=2)
      s[1,1] <- round(qbeta(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qbeta(alf3, priori[1], priori[2]),3)
      v <- rbetabinom.ab(1:1000,size=input$nn,priori[1],priori[2])
      s[2,1] <- as.numeric(quantile(v,alf2))
```

```
      s[2,2] <- as.numeric(quantile(v,alf3))
      s[3,1] <- round(qbeta(alf2, posteriori[1], posteriori[2]),3)
      s[3,2] <- round(qbeta(alf3, posteriori[1], posteriori[2]),3)
      v2                                                          <-
rbetabinom.ab(1:1000,size=input$nsample,posteriori[1],posteriori[2]
)
      s[4,1] <- as.numeric(quantile(v2,alf2))
      s[4,2] <- as.numeric(quantile(v2,alf3))
      rownames(s)                <-                c("Prior","Prior
predictive","Posterior","Posterior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }
  }
})
```

10.11 server-inference_normal.R

```
#StatClip
#Nuria Busquets, February 2016

#server-inference_normal.R

output$normaldistr <- renderUI({
  if(input$mod_distr_inference == "norm"){
    h3(HTML("X ~ N(&mu;, &sigma;)"))
  }
})

output$pm <- renderUI({
  if(input$mod_distr_inference == "norm"){
    div(
      h3(HTML("Prior knowledge about &mu;")),
      h3(HTML("&mu; ~ Normal (m, s)"))
    )
  }
})

output$mu_mean <- renderUI({
  if(input$mod_distr_inference == "norm"){
    sliderInput(inputId="mumean",
                label = "Mean",
                value = 0,
                min = -10,
                max = 10)
  }})

output$mu_sigma <- renderUI({
  if(input$mod_distr_inference == "norm"){
    sliderInput(inputId="musigma",
                label = "Standard deviation ",
                value = 1,
                min = 0.01,
                max = 10)
  }})

output$psig <- renderUI({
  if(input$mod_distr_inference == "norm"){
    div(
      h3(HTML("Prior knowledge about &sigma;")),
      h3(HTML("<sup>1</sup>&frasl;<sub>&sigma;<sup>2</sup></sub>~
Gamma(&alpha;, &beta;)")),
      h3(HTML("&sigma; ~ f(&sigma;)"))
    )
  }
})

output$sig_mean <- renderUI({
  if(input$mod_distr_inference == "norm"){
    sliderInput(inputId="sigmean",
                label = "Mean",
                value = 2,
                min = 0.75,
                max = 6)
  }})

output$sig_sigma <- renderUI({
```

```
  if(input$mod_distr_inference == "norm"){
    sliderInput(inputId="sigsigma",
                label = "Standard deviation",
                value = 0.4,
                min = 0.31,
                step = 0.01,
                max = 0.5*input$sigmean)
  }})

output$htext <- renderPrint({
  if(input$mod_distr_inference == "norm"){
    helpText("Previously you have to insert the data in the section
'Data'")
  }
})

output$selectvar_itp <- renderUI({
  if(input$mod_distr_inference == "norm"){
    selectInput("var_itp",   label="Choose   the   data   variable",
choices=variable_names())
  }
})

output$dat_inference_two_par <- renderUI({
  if(input$mod_distr_inference == "norm"){
    actionButton("up_itp_norm", label = "Update")
  }
})

gibbs1 <- reactive({
  y <- working_data$data[,match(input$var_itp, variable_names())]
  mu_0 <- input$mumean
  l_0 <- 1/(input$musigma^2)
  n <- length(y)
  source("BisectionMethod.R", local=TRUE)
  E <- input$sigmean
  V <- (input$sigsigma)^2
  fa <- function(a){
    (((E^2)*gamma(a)^2)/(gamma(a-0.5)^2))-((a-1)*(V+E^2))
  }
  aa <- bisection_method(fa, 1.05, 90)
  a <- aa[1]
  b <- (a-1)*(V+E^2)
  mu <- 2
  l <- 3
  an <- a+(n/2)
  bn <- b+(1/2)*sum((y-mu)^2)
  l[2] <- rgamma(1, an, bn)
  l_n <- l_0+n*l[1]
  mu_n <-(1/l_n)*(l_0*mu_0+n*l[1]*mean(y))
  mu[2] <- rnorm(1,mean=mu_n,sd=1/sqrt(l_n))
  for(i in 2:13000){
    bn <- b+(1/2)*sum((y-mu[i-1])^2)
    l[i] <- rgamma(1, an, bn)
    l_n <- l_0+n*l[i]
    mu_n <-(1/l_n)*(l_0*mu_0+n*l[i-1]*mean(y))
    mu[i] <- rnorm(1,mean=mu_n,sd=1/sqrt(l_n))
  }
  mu <- mu[-c(1:7000)]
  l <- l[-c(1:7000)]
  mu2 <- mu[-c(which(mu<as.numeric(quantile(mu, 0.05))))]
  mu2 <- mu[-c(which(mu>as.numeric(quantile(mu, 0.95))))]
  l2 <- l[-c(which(l>as.numeric(quantile(l, 0.9))))]
  llista <- list(mu=mu, l=l,mu2=mu2, l2=l2)
  return(llista)
})

gibbs2 <- reactive({
  y <- working_data$data[,match(input$var_itp, variable_names())]
  mu_0 <- input$mumean
  l_0 <- 1/(input$musigma^2)
  n <- length(y)
  source("BisectionMethod.R", local=TRUE)
  E <- input$sigmean
  V <- (input$sigsigma)^2
  fa <- function(a){
    (((E^2)*gamma(a)^2)/(gamma(a-0.5)^2))-((a-1)*(V+E^2))
  }
```

```
  aa <- bisection_method(fa, 1.05, 90)
  a <- aa[1]
  b <- (a-1)*(V+E^2)
  mu <- 200
  l <- 300
  an <- a+(n/2)
  bn <- b+(1/2)*sum((y-mu)^2)
  l[2] <- rgamma(1, an, bn)
  l_n <- l_0+n*l[1]
  mu_n <-(1/l_n)*(l_0*mu_0+n*l[1]*mean(y))
  mu[2] <- rnorm(1,mean=mu_n,sd=1/sqrt(l_n))
  for(i in 2:13000){
    bn <- b+(1/2)*sum((y-mu[i-1])^2)
    l[i] <- rgamma(1, an, bn)
    l_n <- l_0+n*l[i]
    mu_n <-(1/l_n)*(l_0*mu_0+n*l[i-1]*mean(y))
    mu[i] <- rnorm(1,mean=mu_n,sd=1/sqrt(l_n))
  }
  mu <- mu[-c(1:7000)]
  l <- l[-c(1:7000)]
  mu2 <- mu[-c(which(mu<as.numeric(quantile(mu, 0.05))))]
  mu2 <- mu[-c(which(mu>as.numeric(quantile(mu, 0.95))))]
  l2 <- l[-c(which(l>as.numeric(quantile(l, 0.9))))]
  llista <- list(mu=mu, l=l,mu2=mu2, l2=l2)
  return(llista)
})

output$plot_mu_sigma0 <- renderPlot({
  if(input$mod_distr_inference == "norm"){
    source("BisectionMethod.R", local=TRUE)
    E <- input$sigmean
    V <- (input$sigsigma)^2
    fa <- function(a){
      (((E^2)*gamma(a)^2)/(gamma(a-0.5)^2))-((a-1)*(V+E^2))
    }
    aa <- bisection_method(fa, 1.05, 90)
    a <- aa[1]
    b <- (a-1)*(V+E^2)
    prior_sigma <- function(y){
      (2/y^3)*(b^a/gamma(a))*(1/y^2)^(a-1)*exp(-b/y^2)
    }
    par(mfrow=c(1,2))
    curve(dnorm(x,mean=input$mumean,sd=input$musigma),from=-50,
to=50, ylab="",
          xlab = expression(mu),lty=1, col="blue", lwd = 3 )
    title("Prior distribution ")
    legend("topright", "Normal distribution", lty = 1,col="blue",
lwd = 3)
    curve( prior_sigma(x),from=0, to=20, ylab="",
           xlab = expression(sigma), lty=1, col="orange", lwd = 3)
    title("Prior distribution ")
    legend("topright", "Sigma distribution", lty = 1,col="orange",
lwd = 3)
    par(mfrow=c(1,1))
  }
})

output$plot_mu_sigma <- renderUI({
  if(input$mod_distr_inference == "norm"){
    plotOutput("plot_mu_sigma0")}
})

output$slow <- renderPrint({
  if(input$mod_distr_inference == "norm"){
    helpText("Press the 'Update' button and wait
             few seconds to see posterior distribution")
  }
})

output$post_mu_sigma0 <- renderPlot({
  if(input$mod_distr_inference == "norm"){
    if (input$up_itp_norm == 0){
      return()
    }else{
      mu2 <- ((gibbs1())$mu2)
      l2 <- ((gibbs1())$l2)
      par(mfrow=c(1,2))
```

```
      plot(density(mu2,     adjust=2),     ylab="",     xlab     =
expression(mu),lty=1, col="blue", lwd = 3,
           main="Posterior distribution ")
      legend("topright", "Normal distribution", lty = 1,col="blue",
lwd = 3)
      plot(   density(1/sqrt(l2),   adjust=2),   ylab="",   xlab   =
expression(sigma), lty=1, col="orange", lwd = 3,
            main="Posterior distribution")
      legend("topright", "Sigma distribution", lty = 1,col="orange",
lwd = 3)
      par(mfrow=c(1,1))
    }
  }
})

output$post_mu_sigma <- renderUI({
  if(input$mod_distr_inference == "norm"){
    plotOutput("post_mu_sigma0")}
})

output$prior_pred_mu_sigma0 <- renderPlot({
  if(input$mod_distr_inference == "norm"){
    mumean <- input$mumean
    musigma <- input$musigma
    rmu <- rnorm(13000, mean=mumean, sd=musigma)
    source("BisectionMethod.R", local=TRUE)
    E <- input$sigmean
    V <- (input$sigsigma)^2
    fa <- function(a){
      (((E^2)*gamma(a)^2)/(gamma(a-0.5)^2))-((a-1)*(V+E^2))
    }
    aa <- bisection_method(fa, 1.05, 90)
    a <- aa[1]
    b <- (a-1)*(V+E^2)
    rsd = rgamma(13000,a, b)
    y<- rnorm(1,mean=rmu[1], sd=1/sqrt(rsd[1]))
    for( i in 2:13000){
      y[i] <- rnorm(1,mean=rmu[i], sd=1/sqrt(rsd[1]))
    }
    plot(density(y,adjust=2),    xlab="",    ylab="",    main="Prior
predictive distribution ",lty = 1, col="green",lwd= 3)
    legend("topright", "Normal",
           lty = 1, col="green",lwd= 3)
  }
})

output$prior_pred_mu_sigma<- renderUI({
  if(input$mod_distr_inference == "norm"){
    plotOutput("prior_pred_mu_sigma0")}
})

output$post_pred_mu_sigma0 <- renderPlot({
  if(input$mod_distr_inference == "norm"){
    if (input$up_itp_norm == 0){
      return()
    }else{
      mu <- ((gibbs1())$mu2)
      l <- ((gibbs1())$l2)
      ynew <- rnorm(1,mean=mu[1], sd=1/sqrt(l[1]))
      q <- min(length(mu), length(l))
      for( i in 2:q){
        ynew[i] <- rnorm(1,mean=mu[i], sd=1/sqrt(l[i]))
      }
      ynew2     <-     ynew[-c(which(ynew<as.numeric(quantile(ynew,
0.05))))]
      ynew2     <-     ynew[-c(which(ynew>as.numeric(quantile(ynew,
0.95))))]
      plot(density(ynew2,adjust=2), xlab="", ylab="",
           main="Posterior  predictive  distribution  ",lty  =  1,
col="green",
           lwd= 3)
      legend("topright", "Normal",
             lty = 1, col="green",lwd= 3)
    }
  }
})

output$post_pred_mu_sigma<- renderUI({
```

```
  if(input$mod_distr_inference == "norm"){
    plotOutput("post_pred_mu_sigma0")}
})

output$table_normal_1 <- renderTable({
  if(input$mod_distr_inference == "norm"){
    if(input$up_itp_norm == 0){
      sortida <- matrix(nrow = 2, ncol = 3)
      colnames(sortida) <- c("Prior mu", "Prior sigma", "Prior
Predictive")
      rownames(sortida) <- c("Mean", "Variance")
      sortida[1, 1] <- input$mumean
      sortida[2, 1] <- input$musigma^2
      sortida[1, 2] <- input$sigmean
      sortida[2, 2] <- (input$sigsigma)^2
      rmu <- rnorm(13000, mean=input$mumean, sd=input$musigma)
      source("BisectionMethod.R", local=TRUE)
      E <- input$sigmean
      V <- (input$sigsigma)^2
      fa <- function(a){
        (((E^2)*gamma(a)^2)/(gamma(a-0.5)^2))-((a-1)*(V+E^2))
      }
      aa <- bisection_method(fa, 1.05, 90)
      a <- aa[1]
      b <- (a-1)*(V+E^2)
      rsd = rgamma(13000,a, b)
      y<- rnorm(1,mean=rmu[1], sd=1/sqrt(rsd[1]))
      for( i in 2:13000){
        y[i] <- rnorm(1,mean=rmu[i], sd=1/sqrt(rsd[1]))
      }
      sortida[1, 3] <- mean(y)
      sortida[2, 3] <- sd(y)^2
      round(sortida, 3)
    }else{
      sortida <- matrix(nrow = 2, ncol = 6)
      colnames(sortida) <- c("Prior mu", "Posterior mu", "Prior
sigma", "Posterior sigma",
                             "Prior Predictive", "Posterior
Predictive")
      rownames(sortida) <- c("Mean", "Variance")
      sortida[1, 1] <- input$mumean
      sortida[2, 1] <- input$musigma^2
      sortida[1, 3] <- input$sigmean
      sortida[2, 3] <- (input$sigsigma)^2
      rmu <- rnorm(13000, mean=input$mumean, sd=input$musigma)
      source("BisectionMethod.R", local=TRUE)
      E <- input$sigmean
      V <- (input$sigsigma)^2
      fa <- function(a){
        (((E^2)*gamma(a)^2)/(gamma(a-0.5)^2))-((a-1)*(V+E^2))
      }
      aa <- bisection_method(fa, 1.05, 90)
      a <- aa[1]
      b <- (a-1)*(V+E^2)
      rsd = rgamma(13000,a, b)
      y<- rnorm(1,mean=rmu[1], sd=1/sqrt(rsd[1]))
      for( i in 2:13000){
        y[i] <- rnorm(1,mean=rmu[i], sd=1/sqrt(rsd[1]))
      }
      sortida[1, 5] <- mean(y)
      sortida[2, 5] <- sd(y)^2
      mu_post <- ((gibbs1())$mu2)
      l_post <- ((gibbs1())$l2)
      sortida[1, 2] <- mean(mu_post)
      sortida[2, 2] <- sd(mu_post)^2
      sortida[1, 4] <- mean(1/sqrt(l_post))
      sortida[2, 4] <- sd(1/sqrt(l_post))^2
      ynew <- rnorm(1,mean= mu_post[1], sd=1/sqrt(l_post[1]))
      q <- min(length(mu_post), length(l_post))
      for( i in 2:q){
        ynew[i] <- rnorm(1,mean=mu_post[i], sd=1/sqrt(l_post[i]))
      }
      ynew2     <-     ynew[-c(which(ynew<as.numeric(quantile(ynew,
0.05))))]
      ynew2     <-     ynew[-c(which(ynew>as.numeric(quantile(ynew,
0.95))))]
      sortida[1, 6] <- mean(ynew2)
      sortida[2, 6] <- sd(ynew2)^2
```

```
      round(sortida, 3)
    }
  }
})

output$convergencia0 <- renderPlot({
  if(input$mod_distr_inference == "norm"){
    if(input$up_itp_norm == 0){ return()
    }else{
      c1mu <- ((gibbs1())$mu)
      c1l  <- ((gibbs1())$l)
      c2mu <- ((gibbs2())$mu)
      c2l  <- ((gibbs2())$l)
      par(mfrow=c(2,1))
      plot.ts(c1mu, main="mu", ylab=" ")
      lines(c2mu, col="orange")
      plot.ts(c1l, main="l", ylab=" ")
      lines(c2l, col="orange")
      par(mfrow=c(1,1))
    }
  }
})

output$convergencia<- renderUI({
  if(input$mod_distr_inference == "norm"){
    plotOutput("convergencia0")}
})

output$Rhat <- renderTable({
  if(input$mod_distr_inference == "norm"){
    if(input$up_itp_norm == 0){ return()}
    else{
      c1mu <- ((gibbs1())$mu)
      c1l  <- ((gibbs1())$l)
      c2mu <- ((gibbs2())$mu)
      c2l  <- ((gibbs2())$l)
      sup <- seq(from=1000, to=6000, by=1000)
      l <- length(sup)-1
      rhat_mu <- 0
      rhat_l <- 0
      for( i in 1:l){
        v1 <- var(c1mu[c(sup[i]:sup[i+1])])
        v2 <- var(c2mu[c(sup[i]:sup[i+1])])
        vt            <-            var(c(c1mu[c(sup[i]:sup[i+1])],
c2mu[c(sup[i]:sup[i+1])]))
        rhat_mu[i] <- vt/((v1+v2)/2)
        v1 <- var(c1l[c(sup[i]:sup[i+1])])
        v2 <- var(c2l[c(sup[i]:sup[i+1])])
        vt             <-             var(c(c1l[c(sup[i]:sup[i+1])],
c2l[c(sup[i]:sup[i+1])]))
        rhat_l[i] <- vt/((v1+v2)/2)
      }
      f <- length(rhat_mu)
      sortida <- matrix(nrow = f, ncol = 2)
      colnames(sortida) <- c("mu", "l")
      rownames(sortida) <- c("1000-2000", "2000-3000", "3000-4000",
"4000-5000", "5000-6000")
      for(i in 1:f){
        sortida[i,1] <- rhat_mu[i]
        sortida[i,2] <- rhat_l[i]
      }
      return(sortida)
    }}
})
```

10.12 server-inference_poisson.R

```
#StatClip
#Nuria Busquets, February 2016

#server-inference_poisson.R

output$prior_poisson <- renderUI({
  if(input$mod_distr_inference == "poi"){
    div(
      h3("Prior distribution:"),
      h3(HTML("&theta; ~ Gamma(&alpha;, &beta;)")),
```

```
      h3("Priror predictive distribution:"),
      h3(HTML("P <sub>&pi;</sub> (y) ~ Negative Binomial(&alpha;,
<sup>&beta;</sup>&frasl;<sub>(&beta;+1)</sub>)"))
    )
  }
})

output$mean_poisson <- renderUI({
  if(input$mod_distr_inference == "poi"){
    sliderInput(inputId="mu_p",
                label = "Mean",
                value = 2,
                min = 0.1,
                max = 20)
  }
})

output$sd_poisson <- renderUI({
  if(input$mod_distr_inference == "poi"){
       sliderInput(inputId="sd_p",
                label="Standard deviation",
                value = 1,
                min=0.1,
                max=10)
  }})

output$htext_poi_data <- renderPrint({
  if(input$mod_distr_inference == "poi"){
    div(
      h3("Data"),
      helpText("Previously you have to insert the data in the section
'Data'")
    )
  }
})

output$data_poi <- renderUI({
  if(input$mod_distr_inference == "poi"){
    selectInput("y_poi", label="Choose the data variable",
                choices=variable_names())
  }
})

output$data_poi2<- renderUI({
  if(input$mod_distr_inference == "poi"){
    actionButton("up_poi", label = "Update")
  }
})

pois_param <- reactive({
  a <- (input$mu_p^2)/(input$sd_p^2)
  b <- (input$mu_p)/(input$sd_p^2)
  llista <- list(a=a, b=b)
})

output$dpriori_poisson0 <- renderPlot({
    a <- (pois_param()$a)
    b <- (pois_param()$b)
    curve(dgamma(x, a,b),from=0, to=30,lty=1, col="blue", lwd = 3,
ylab="",
          xlab = expression(theta))
    title("Prior distribution ")
    legend("topright", "Gamma distribution", lty = 1,col="blue", lwd
= 3)
})

output$dpriori_poisson <- renderUI({
  if(input$mod_distr_inference == "poi"){
    plotOutput("dpriori_poisson0")}
})

output$cg_poi <- renderUI({
  if(input$mod_distr_inference == "poi"){
    if (input$up_poi == 0){
      return()
    }
    isolate(
```

```
      checkboxGroupInput("checkGroup_poi",
                         label   =   h3("Posterior   distribution's
graphic"),
                         choices = list("Prior" = 1, "Likelihood" =
2),
                         selected = c(1,2)
      )
    )
  }
})

output$dprioripostvar_poi0 <- renderPlot({
  if(input$mod_distr_inference == "poi"){
    if (input$up_poi == 0){
      return()
    }
  }
  isolate({
    suport <- seq(from = 0, to = 100, length = 1000)
    y <- working_data$data[,match(input$y_poi, variable_names())]
    n <- length(y)
    a <- (pois_param()$a)
    b <- (pois_param()$b)
    dist.priori  <- dgamma(suport, a,b)
    posteriori <- c(a + sum(y), b+n)
    dist.posteriori <- dgamma(suport, posteriori[1], posteriori[2])
    vers <- rep(0, length(suport))
    Int.vers <- 0
    for (i in 1:length(suport)) {
      vers[i] <- (suport[i]^sum(y))*exp(-n*suport[i])
    }
    K <- integrate(function(x)(exp(sum(y)*log(x)-n*x)),
                   lower      =      0.01,      upper      =      99,
subdivisions=1000)$value
    vers.resc <- vers/K
    })
  plot(suport,dist.posteriori,                                ylab="",
xlab=expression(theta),type="l",
       lty=1, col="red", lwd = 3)
  if(sum(as.numeric(input$checkGroup_poi))==1){
    lines(suport, dist.priori, lty=1, col="blue", lwd = 3)}
  if(sum(as.numeric(input$checkGroup_poi))==2){
    lines(suport, vers.resc, lty=3, col="black", lwd = 2)}
  if(sum(as.numeric(input$checkGroup_poi))==3){
    lines(suport, dist.priori,lty=1, col="blue", lwd = 3)
    lines(suport, vers.resc, lty=3, col="black", lwd = 2)
  }
  title("Posterior distribution ")
  legend("topright", c("posterior","prior","likelihood"),
         lty = c(1,1,3), col=c("red","blue","black"), lwd= c(3,3,2))
})

output$dprioripostvar_poi <- renderUI({
  if(input$mod_distr_inference == "poi"){
    plotOutput("dprioripostvar_poi0")}
})

output$dpredpriori_poi0 <- renderPlot({
  a <- (pois_param()$a)
  b <- (pois_param()$b)
  pre.prior <- dnbinom(1:100, a, b/(1+b))
  plot(1:100,  pre.prior,  ylab  =  "",  xlab  =  "",  type  =  "h",
col="blue")
  title("Prior predictive distribution \n Negative binomial ")
})

output$dpredpriori_poi <- renderUI({
  if(input$mod_distr_inference == "poi"){
    plotOutput("dpredpriori_poi0")}
})

output$dpredpost_poi0 <- renderPlot({
  if(input$mod_distr_inference == "poi"){
    if (input$up_poi == 0){
      return()
    }
  }
  isolate({
```

```
    a <- (pois_param()$a)
    b <- (pois_param()$b)
    y <- working_data$data[,match(input$y_poi, variable_names())]
    n <- length(y)
    posteriori <- c(a + sum(y), b + n)
    pre.post        <-        dnbinom(1:100,        posteriori[1],
posteriori[2]/(1+posteriori[2]))
    plot(1:100,  pre.post,  ylab  =  "",  xlab  =  "",  type  =  "h",
col="blue")
    title("Posterior predictive distribution \n Negative binomial ")
  })
})

output$dpredpost_poi <- renderUI({
  if(input$mod_distr_inference == "poi"){
    plotOutput("dpredpost_poi0")}
})

output$table1_poi <- renderTable({
  if(input$mod_distr_inference == "poi"){
    if(input$up_poi == 0){
      sortida <- matrix(nrow = 5, ncol = 1)
      colnames(sortida) <- c("Prior")
      rownames(sortida)  <-  c("Alpha","Beta",  "Mean",  "Variance",
"Median")
      a <- (pois_param()$a)
      b <- (pois_param()$b)
      priori <- c(a,b)
      sortida[1, 1] <- priori[1]
      sortida[2, 1] <- priori[2]
      sortida[3, 1] <- input$mu_p
      sortida[4, 1] <- input$sd_p^2
      sortida[5, 1] <- qgamma(0.5, priori[1], priori[2])
      return(sortida)
    }else{
      a <- (pois_param()$a)
      b <- (pois_param()$b)
      y <- working_data$data[,match(input$y_poi, variable_names())]
      n <- length(y)
      priori <- c(a,b)
      posteriori <- c(a + sum(y), b + n)
      sortida <- matrix(nrow = 5, ncol = 2)
      sortida[1, 1] <- priori[1]
      sortida[2, 1] <- priori[2]
      sortida[3, 1] <- input$mu_p
      sortida[4, 1] <- input$sd_p^2
      sortida[5, 1] <- qgamma(0.5, priori[1], priori[2])
      sortida[1, 2] <- posteriori[1]
      sortida[2, 2] <- posteriori[2]
      sortida[3, 2] <- posteriori[1]/posteriori[2]
      sortida[4, 2] <- posteriori[1]/(posteriori[2]^2)
      sortida[5, 2] <- qgamma(0.5, posteriori[1], posteriori[2])
      colnames(sortida) <- c("Prior","Posterior")
      rownames(sortida)  <-  c("Alpha","Beta",  "Mean",  "Variance",
"Median")
      return(round(sortida, 3))
    }
  }
})

output$IC1_poi <- renderUI({
  if(input$mod_distr_inference == "poi"){
    div(
      column(width=4,
             br(),
             h5("Introduce level of credibility")),
      column(width=2,
             numericInput("icl1_poi",label="",value   =   95,min=0,
max=100)),
      column(width=1,
             br(),
             h5("%"))
    )
  }
})

output$table2_poi <- renderTable({
  if(input$mod_distr_inference == "poi"){
```

```
    if(input$up_poi == 0){
      l <- input$icl1_poi/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- (pois_param()$a)
      b <- (pois_param()$b)
      priori <- c(a,b)
      s <- matrix(nrow=2,ncol=2)
      s[1,1] <- round(qgamma(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qgamma(alf3, priori[1], priori[2]),3)
      s[2,1]        <-        round(qnbinom(alf2,        priori[1],
priori[2]/(1+priori[2])),3)
      s[2,2]        <-        round(qnbinom(alf3,        priori[1],
priori[2]/(1+priori[2])),3)
      rownames(s) <- c("Prior","Prior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }else{
      l <- input$icl1_poi/100
      alf <- 1-l
      alf2 <- alf/2
      alf3 <- (1-alf2)
      a <- (pois_param()$a)
      b <- (pois_param()$b)
      y <- working_data$data[,match(input$y_poi, variable_names())]
      n <- length(y)
      ap  <-  a + sum(y)
      bp  <-  b + n
      priori <- c(a,b)
      posteriori <- c(ap,bp)
      s <- matrix(nrow=4,ncol=2)
      s[1,1] <- round(qgamma(alf2, priori[1], priori[2]),3)
      s[1,2] <- round(qgamma(alf3, priori[1], priori[2]),3)
      s[2,1]        <-        round(qnbinom(alf2,        priori[1],
priori[2]/(1+priori[2])),3)
      s[2,2]        <-        round(qnbinom(alf3,        priori[1],
priori[2]/(1+priori[2])),3)
      s[3,1] <- round(qgamma(alf2, posteriori[1], posteriori[2]),3)
      s[3,2] <- round(qgamma(alf3, posteriori[1], posteriori[2]),3)
      s[4,1]       <-       round(qnbinom(alf2,       posteriori[1],
posteriori[2]/(1+posteriori[2])),3)
      s[4,2]       <-       round(qnbinom(alf3,       posteriori[1],
posteriori[2]/(1+posteriori[2])),3)
      rownames(s)                <-                c("Prior","Prior
predictive","Posterior","Posterior predictive")
      colnames(s) <- c("Lower bound", "Upper bound")
      return(s)
    }
  }
})
```

10.13 server-load_data_set.R

```
#StatClip
#Eduard Serrahima, May 2015

#server-load_data_set.R
library(DT)
#Serverfunctions for the Load Data Set tab
#file_data is a reactive functions that stores the data frame read
from the
#file updated
file_data <- reactive({
  inFile <- input$file
  if (is.null(inFile)){
    return(NULL)
  }
  data       <-read.csv(inFile$datapath,       header=input$header,
sep=input$sep,dec = ",",
           quote=input$quote)
  return(data)
})

#sample_data is a reactive function that stores the data frame
#from the sample data sets once the corresponding button is pressed
sample_data_iris <- reactive({
```

```
  data <- NULL
  if(input$iris >=1){
    try(data <- iris)
  }
})

sample_data_mtcars <- reactive({
  data <-NULL
  if(input$mtcars >= 1){
    try(data<-mtcars)
  }
})

table_data <- reactiveValues(which=NULL)
observeEvent(input$file, {
  table_data$which <- "upfile"
})
observeEvent(input$iris, {
  table_data$which <- "sample_iris"
})
observeEvent(input$mtcars, {
  table_data$which <- "sample_mtcars"
})

#Loaded data
loaded_data <- reactive({
  if(is.null(file_data())    &    is.null(sample_data_iris())    &
is.null(sample_data_mtcars())){
    return(NULL)
  }
  else if(table_data$which == "upfile"){
    return(file_data())
  }
  else if(table_data$which == "sample_iris"){
    return(sample_data_iris())
  }
  else if(table_data$which == "sample_mtcars"){
    return(sample_data_mtcars())
  }
})

#Funcion that creates the data table output
datatable <- DT::renderDataTable({
  if (is.null(loaded_data())){
    return(NULL)
  }
  else{
    DT::datatable(loaded_data())
  }
})
```

10.14 server-simple_linear_regression.R

```
#StatClip
#Nuria Busquets, February 2016

#server-simple_linear_regression.R

output$prior_beta0 <- renderUI({
  if(input$priorknowledge==1){
    div(
      fluidRow(
        column(width=12,
               h3(HTML("Knowledge about &beta;<sub>0</sub>"))
        )
      ),
      column(width =6,
        numericInput("mean_b0", label = h4("Mean"), value = 1)
      ),
      column(width =6,
        numericInput("sd_b0",  label  =  h4("Standard  deviation"),
value = 1)
      )
    )
```

```
  }else{
    return()
  }
})

output$prior_beta1 <- renderUI({
  if(input$priorknowledge==1){
  div(
    fluidRow(
      column(width=12,
             h3(HTML("Knowledge about &beta;<sub>1</sub>"))
      )
    ),
    column(width =6,
      numericInput("mean_b1", label = h4("Mean"), value = 1)
    ),
    column(width =6,
      numericInput("sd_b1", label = h4("Standard deviation"), value
= 1)
    )
  )
  }else{
    return()
  }
})

selectysimplelinearregression <- renderUI({
  selectInput("y_slr",
              label="Choose the Y variable for the linear
regression",
              choices=variable_names(),
              )
})

selectxsimplelinearregression <- renderUI({
  selectInput("x_slr",
              label="Choose the X variable for the linear
regression",
              choices=variable_names()[-
which(variable_names()==input$y_slr)],
              )
})

output$m <- renderTable({
  if(input$up_slr== 0){
    return()
  }
  isolate({
    if(input$priorknowledge==2){
      x <- match(input$x_slr, variable_names())
      y <- match(input$y_slr, variable_names())
      mydata <- working_data$data[,c(x,y)]
      colnames(mydata) <- c('x','y')
      model <- MCMCglmm(y~x, data=mydata)
      m <- summary(model)
      s <- matrix(ncol=5, nrow=2)
      rownames(s) <- c('Intercept',"X")
      colnames(s) <- c('Post. mean', 'l-95% CI', 'u-95% CI',
'eff.samp', 'pMCMC')
      s[1,1] <- m$solutions[1]
      s[2,1] <- m$solutions[2]
      s[1,2] <- m$solutions[3]
      s[2,2] <- m$solutions[4]
      s[1,3] <- m$solutions[5]
      s[2,3] <- m$solutions[6]
      s[1,4] <- m$solutions[7]
      s[2,4] <- m$solutions[8]
      s[1,5] <- m$solutions[9]
      s[2,5] <- m$solutions[10]
      return(s)
    }else{
      x <- match(input$x_slr, variable_names())
      y <- match(input$y_slr, variable_names())
      mydata <- working_data$data[,c(x,y)]
      colnames(mydata) <- c('x','y')
      v <- diag(c((input$sd_b0)^2,(input$sd_b1)^2))
      m <- c(input$mean_b0,input$mean_b1)
```

```
      B <- list(V=v, mu=m)
      model <- MCMCglmm(y~x, data=mydata,prior=list(B=B))
      m <- summary(model)
      s <- matrix(ncol=5, nrow=2)
      rownames(s) <- c("Intercept","X")
      colnames(s)  <-  c("Post.  mean",  "l-95%  CI",  "u-95%  CI",
"eff.samp", "pMCMC")
      s[1,1] <- m$solutions[1]
      s[2,1] <- m$solutions[2]
      s[1,2] <- m$solutions[3]
      s[2,2] <- m$solutions[4]
      s[1,3] <- m$solutions[5]
      s[2,3] <- m$solutions[6]
      s[1,4] <- m$solutions[7]
      s[2,4] <- m$solutions[8]
      s[1,5] <- m$solutions[9]
      s[2,5] <- m$solutions[10]
      return(s)
    }
  })
})

output$title_slr_plot <- renderPrint({
  h4("Posterior distribution of parameters")
})

output$postgrafmodel0 <- renderPlot({
  if(input$up_slr== 0){
    return()
  }
  isolate({
    if(input$priorknowledge==2){
      x <- match(input$x_slr, variable_names())
      y <- match(input$y_slr, variable_names())
      mydata <- working_data$data[,c(x,y)]
      colnames(mydata) <- c('x','y')
      model <- MCMCglmm(y~x, data=mydata)
      plot(model$Sol)
    }else{
      x <- match(input$x_slr, variable_names())
      y <- match(input$y_slr, variable_names())
      mydata <- working_data$data[,c(x,y)]
      colnames(mydata) <- c('x','y')
      v <- diag(c((input$sd_b0)^2,(input$sd_b1)^2))
      m <- c(input$mean_b0,input$mean_b1)
      B <- list(V=v, mu=m)
      model <- MCMCglmm(y~x, data=mydata,prior=list(B=B))
      plot(model$Sol)
    }
  })
})

output$postgrafmodel <- renderUI({
  plotOutput("postgrafmodel0")
})


output$title_slr_plot <- renderPrint({
  if(input$up_slr== 0){
    h3("Please, enter the data and press 'Update' button")
  }
  else{
    h3("Estimated simple linear regression")
  }
})

output$postmodel0 <- renderPlot({
  if(input$up_slr== 0){
    return()
  }
  isolate({
    if(input$priorknowledge==2){
      x <- match(input$x_slr, variable_names())
      y <- match(input$y_slr, variable_names())
      mydata <- working_data$data[,c(x,y)]
      colnames(mydata) <- c("x","y")
      model <- MCMCglmm(y~x, data=mydata)
      m <- summary(model)
```

```
      b0 <- m$solution[1]
      b1 <- m$solution[2]
      mm <- function(y) {
        b0+b1*y
      }
      lx<- min(mydata$x,na.rm=T)*0.1
      ux <- max(mydata$x,na.rm=T)*1.3
      ly <- min(mydata$y, na.rm=T)*0.1
      uy <- max(mydata$y,na.rm=T)*1.3
      curve(mm,lty=1, col="red", lwd = 3, from=lx, to=ux, ylim=c(ly,
uy))
      points(mydata$x,mydata$y,pch=16,col="blue")
    }else{
      x <- match(input$x_slr, variable_names())
      y <- match(input$y_slr, variable_names())
      mydata <- working_data$data[,c(x,y)]
      colnames(mydata) <- c("x","y")
      v <- diag(c((input$sd_b0)^2,(input$sd_b1)^2))
      m <- c(input$mean_b0,input$mean_b1)
      B <- list(V=v, mu=m)
      model <- MCMCglmm(y~x, data=mydata,prior=list(B=B))
      m <- summary(model)
      b0 <- m$solution[1]
      b1 <- m$solution[2]
      mm <- function(y) {
        b0+b1*y
      }
      lx<- min(mydata$x,na.rm=T)*0.1
      ux <- max(mydata$x,na.rm=T)*1.3
      ly <- min(mydata$y, na.rm=T)*0.1
      uy <- max(mydata$y,na.rm=T)*1.3
      curve(mm,lty=1, col="red", lwd = 3, from=lx, to=ux, ylim=c(ly,
uy))
      points(mydata$x,mydata$y,pch=16,col="blue")
    }
  })
})

output$postmodel <- renderUI({
  plotOutput("postmodel0")
})
```

10.15 BisectionMethod.R

```
bisection_method <- function (f, a, b, num = 10, eps = 1e-05)
{
  h = abs(b - a)/num
  i = 0
  j = 0
  a1 = b1 = 0
  p=1
  s=0
  while (i <= num) {
    a1 = a + i * h
    b1 = a1 + h
    if (f(a1) == 0) {
      #print(a1)
      #print(f(a1))
      s[p] <- a1
      p <- p+1
      s[p] <- f(a1)
      p <- p+1
    }
    else if (f(b1) == 0) {
      #print(b1)
      #print(f(b1))
      s[p] <- b1
      p <- p+1
      s[p] <- f(b1)
      p <- p+1
    }
    else if (f(a1) * f(b1) < 0) {
      repeat {
        if (abs(b1 - a1) < eps)
          break
        x <- (a1 + b1)/2
        if (f(a1) * f(x) < 0)
```

```
          b1 <- x
        else a1 <- x
      }
      j = j + 1
      s[p] <- (a1 + b1)/2
      p <- p+1
      s[p] <- f((a1 + b1)/2)
      p <- p+1
    }
    i = i + 1
  }
  if (j == 0) {
    print("finding root is fail")
  }
  else print("finding root is successful")
  return(s)
}
```

10.16 ChangePointFunctions.R

```
r.aus <- function(r,p1,p2,y){
  if(r<n){
    aus1 <- p1^(sum(y[1:r])+a-1)
    aus2 <- (1-p1)^(sum(m[1:r])-sum(y[1:r])+b-1)
    aus3 <- p2^(sum(y[(r+1):n])+alpha-1)
    aus4 <- (1-p2)^(sum(m[(r+1):n])-sum(y[(r+1):n])+beta-1)
    out <- aus1*aus2*aus3*aus4
  }else{
    aus1 <- p1^(sum(y[1:n])+a-1)
    aus2 <- (1-p1)^(sum(m[1:n])-sum(y[1:n])+b-1)
    out <- aus1*aus2
  }
  return(out)
}

r.fc <- function(p1,p2,y){
  vecc <- rep(NA,n)
  for(i in 1:n){
    j <- r.aus(i,p1,p2,y)
    vecc[i] <- j*prob_rp[i]
  }
  out <- vecc/sum(vecc)
  return(out)
}
```