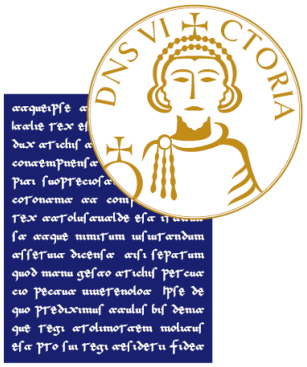


Università degli Studi del Sannio

Dipartimento di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

Data Science

Homework 1

Prof.:
Antonio Pecchia

Studenti:
Stocchetti Federico, matr. 399000543
Fioretti Gabriele, matr. 399000522
Razzano Federica, matr. 399000542

Anno Accademico 2022-2023

Indice

1 Introduzione e Problem Statement

La Prima Parte del progetto si focalizza sulla creazione delle tuple di eventi relativi alle failure del sistema. Il primo passaggio comprende il conteggio delle failure e la selezione delle CWIN, ovvero le finestre di coalescenza, per poi passare al raggruppamento delle entries con la CWIN scelta.

Successivamente, è stata effettuata un'analisi univariata approfondita degli interarrivi. Sono stati esaminati vari aspetti, quali la tendenza centrale, la dispersione e la distribuzione. Sono stati determinati, inoltre, gli intervalli di confidenza. La visualizzazione dei dati è stata effettuata attraverso istogrammi, grafici di densità e boxplot.

A seguire sono stati effettuati la modellazione della reliability utilizzando vari modelli, tra cui quello esponenziale, di Weibull e iperesponenziale e il confronto delle reliability ottenute. La selezione del modello finale è stata basata sui risultati del KS test. Sono state inoltre effettuate analisi aggiuntive sui residui per valutare la bontà dei modelli prodotti.

Nella sezione successiva del report è stato effettuato un lavoro specifico per ogni file di log. Sono stati identificati i set di nodi che tendono presentare una certa correlazione nei loro fallimenti. Per farlo, è stato utilizzato il concetto di itemsets frequenti: per ogni tupla è stata generata una "transazione", costituita dai nodi che appartengono alla tupla stessa. A partire dalle transazioni è stata effettuata l'analisi sui frequent itemset.

Nella sezione finale l'attenzione è stata spostata ai singoli nodi. Dopo aver unito i file in un unico set ordinato di dati, sono stati identificati i 20 nodi più inclini al fallimento. Di questi ne sono stati selezionati i primi 6, salvandone le ricorrenze in file separati.
Successivamente, è stata eseguita una serie di operazioni su questi nodi, tra cui la selezione della CWIN (verificando se questa risultasse la stessa per tutti i nodi) e l'ordinamento in base al MTTF decrescente. Per tutti i sei nodi è stata effettuata la modellazione della reliability, utilizzando i modelli esponenziale, weibull e iperesponenziale.
Infine è stato creato un plot della reliability empirica e dei modelli, eseguito il KS test, ed effettuato analisi aggiuntive per accompagnare i risultati ottenuti.

1.1 Pipeline

In questa sezione viene illustrata la pipeline eseguita per la prima e la seconda parte dell'Homework, indicando file di input e di output e tutti gli script utilizzati.

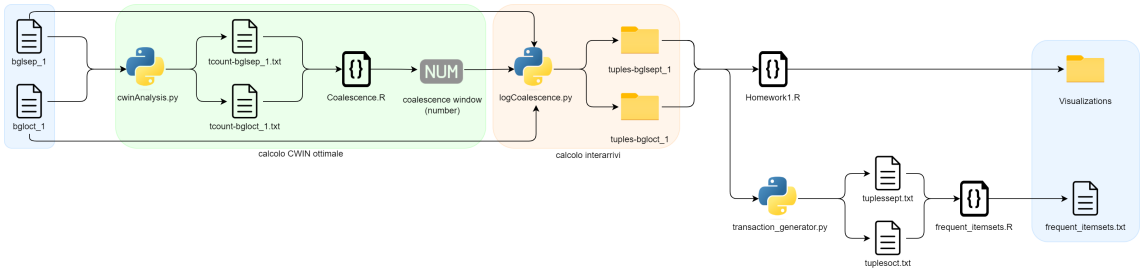


Figura 1: Pipeline Homework Pt.1

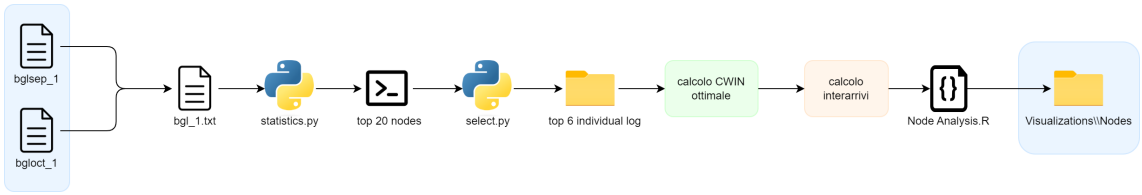


Figura 2: Pipeline Homework Pt.2

2 Analisi sui mesi di Settembre e Ottobre

In questa sezione è descritta l'esecuzione della prima parte dell'Homework, ovvero l'analisi condotta sugli eventi relativi alle failure del sistema.

Verranno effettuati il conteggio delle tuple, la selezione delle CWIN e il rappruppamento delle entries con la CWIN scelta; verrà, inoltre, effettuata una dettagliata analisi univariata degli interarrivi. A seguire verranno eeseguiti il reliability modeling, i plot della reliability e dei modelli ottenuti, KS test, selezione del modello migliore e confronto delle reliability ottenute.
A conclusione della prima parte dell'Homework verranno identificati i set di nodi con la maggiore tendenza a presentare una certa correlazione nei loro fallimenti, tramite il concetto di *transazione*.

2.1 Analisi della CWIN

La prima cosa che è stata fatta per poter analizzare in modo ottimale i log relativi ai mesi di ottobre e settembre, ovvero i files bgloct_1 e bglsep_1, è stato determinare quali fossero le finestre temporali da utilizzare per raggruppare i log al fine di racchiudere in finestre temporali diverse i log relativi ad errori diversi.
Per tale scopo, su ognuno dei files da analizzare è stato eseguito lo script **cwinAnalysis.py** per poter vedere in quante tuple viene diviso il file in analisi in base alla dimensione delle CWIN scelta.
La sintassi del comando usato per eseguire lo script è la seguente:

```
python cwinAnalysis.py <file>
```

Nel nostro caso:

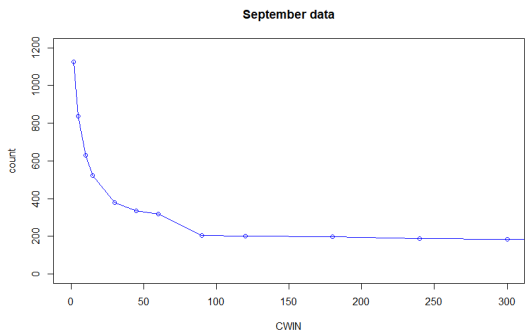
```
python cwinAnalysis.py bgloct_1
python cwinAnalysis.py bglsep_1
```

In seguito all'esecuzione di questi comandi, nella cartella **counts**, sono stati generati due files: **tcount-bgloct_1.txt** e **tcount-bglsep_1.txt**, contenenti l'output atteso.

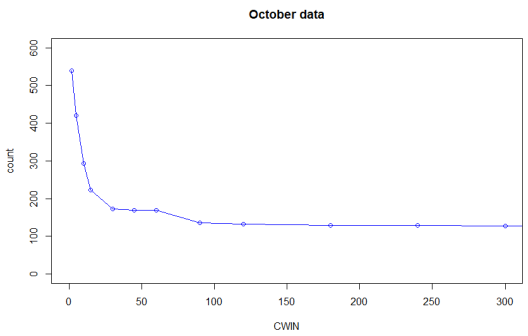
Per poter analizzare questi dati e determinare quale fosse il valore di CWIN da adottare, sono stati plottati i valori contenuti nei file tcount-¡FILE¿.txt utilizzando il seguente codice in R:

```
data_path = paste(getwd(), "/counts/tcount-bglsep_1.txt", sep="")
data<-read.delim(data_path)
plot(data$CWIN,data$COUNT,xlab = "CWIN",ylab = "count",col="blue",type = "ol",
xlim = c(0,300),ylim = c(0,1200))
title(main="CWIN")
```

Utilizzando questo codice su entrambi i files ottenuti dal passaggio precedente, abbiamo ottenuto i seguenti grafici:



(a) September CWIN



(b) October CWIN

Analizzando i grafici, possiamo notare come un valore ottimale della CWIN possa essere **100**, poiché da quel valore in poi la variazione della dimensione di tuple raggruppate CWIN smette di variare significativamente.

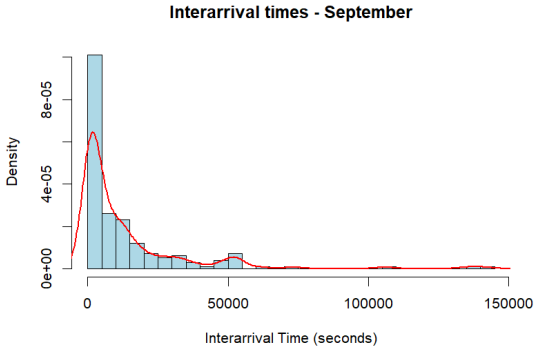
2.2 Analisi degli interarrivi

Tramite lo script **logCoalescence.py** è possibile selezionare le tuple in accordo ad una certa finestra di coalescenza, il cui valore è deciso nella parte precedente. Lo script crea una cartella di output in cui sono presenti tutte le tuple individuate e un file, interarrivals.txt, che rappresenta il tempo che intercorre tra una tupla di eventi e la successiva. Lo script è stato usato sui dati di Settembre e Ottobre, con finestre di coalescenza di, rispettivamente, 120 e 100 secondi. L'analisi univariata degli interarrivi fornisce informazioni rilevanti su come si presentano le failure nel sistema.

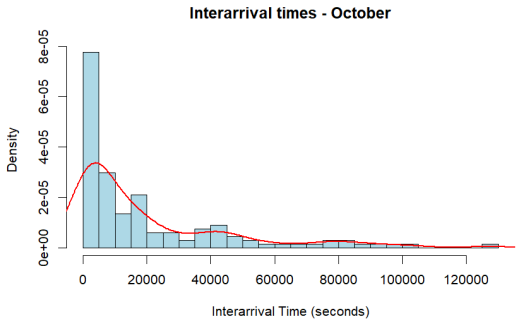
	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
September	139	1192	4676	12668	14604	141067
October	115	2157	8835	19697	26328	126877

Tabella 1: Summary for September and October

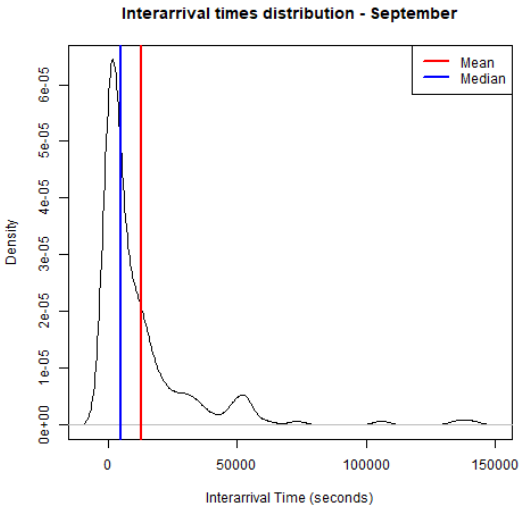
Osserviamo che c'è una forte discrepanza tra il valore massimo e la mediana, oltre che la media e la mediana, a significare la presenza di outliers.



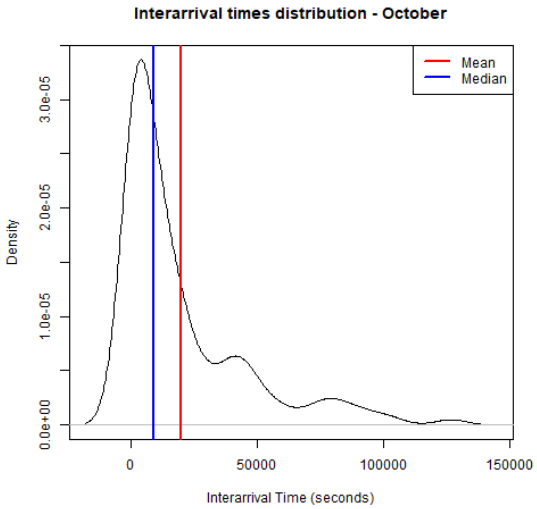
(a) Interarrival times - September distribution



(b) Interarrival times - October distribution

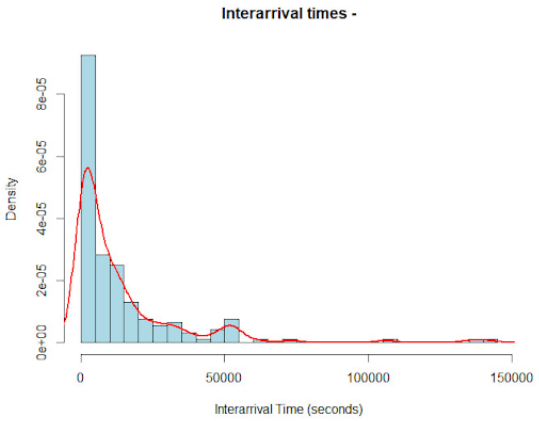


(a) September - central tendency indicators

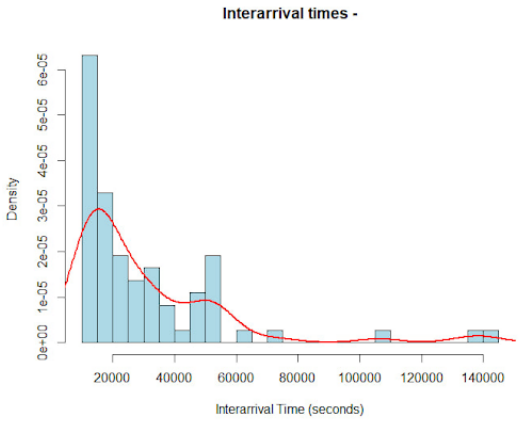


(b) October - central tendency indicators

La distribuzione degli interarrivi ha un picco forte per i valori più bassi, vicini alla finestra di coalescenza, e mostra una coda di valori molto estesa. La distribuzione non è quindi una distribuzione normale, ma una distribuzione "positively skewed". Da notare è che questa distribuzione di valori si mantiene anche variando la finestra di coalescenza e alzandola di molto. Di seguito le immagini con la finestra di coalescenza a 10000 e a 300.



(a) Interarrival times distribution - CWIN 10000



(b) Interarrival times distribution - CWIN 300

Per via della forma non gaussiana della distribuzione il calcolo dell'intervallo di confidenza deve essere svolto in maniera leggermente diversa. Facendo uso della procedura di bootstrapping possiamo avere una misura dell'intervallo di confidenza indiretta, effettuata su più sample del dataset di partenza invece che direttamente sui dati. Questo ci porta ad avere una distribuzione gaussiana nei sample per via del teorema del limite centrale e quindi a poter applicare le formule della gaussiana.

La procedura di bootstrapping è stata effettuata nel codice grazie all'utilizzo della libreria "boot", e sono stati calcolati gli intervalli di confidenza al 90 e al 95

	Settembre	Ottobre
Media degli interarrivi	12647.38 ± 2405.57	19698.83 ± 3618.93
Deviazione standard degli interarrivi	20551.13	25720.47
Intervallo di confidenza al 90%	[10348.86, 15159.99]	[16201.15, 23438.99]
Intervallo di confidenza al 95%	[9991.65, 15654.92]	[15597.97, 24230.14]

Tabella 2: Risultati per i mesi di settembre e ottobre.

Al termine di questa analisi univariata è stato prodotto un boxplot comparativo dei due datasets, in cui è facile notare come le failure di ottobre hanno una dispersione dei valori dalla media più alta, oltre che outliers meno estremi.

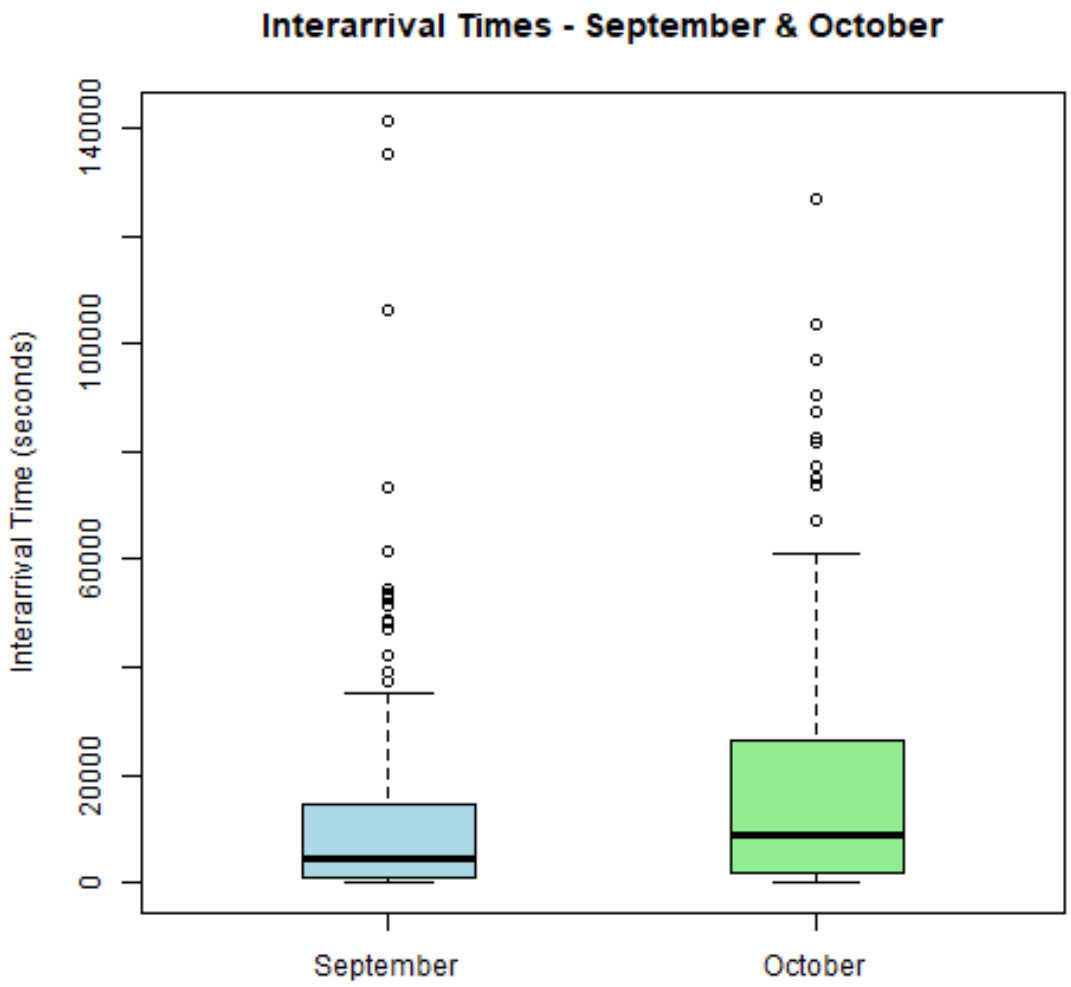
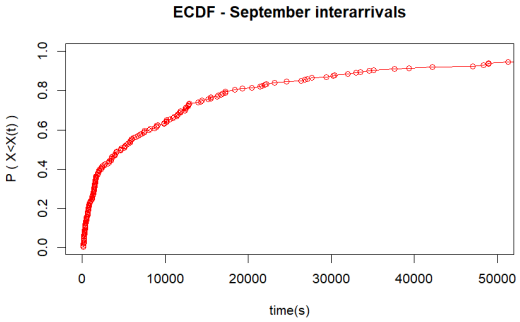


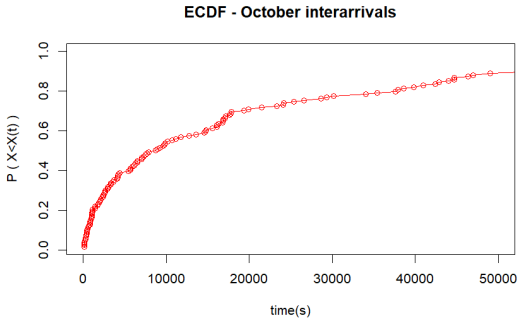
Figura 7: Interarrival Times boxplot comparison - September & October

2.3 Reliability modelling

In questa parte dell'homework sono utilizzati i dati degli interarrivi di Settembre e Ottobre per modellare la reliability del sistema in esame. In particolare, il file interarrivals.txt contiene un dato univariato che rappresenta in secondi la finestra di tempo tra una tupla di failure e la successiva. Questo significa che il dato rappresenta il time to failure (TTF). Possiamo rappresentare una CDF empirica del TTF per sapere quale è la probabilità che si verifichi una failure nel sistema dopo un certo numero di secondi.

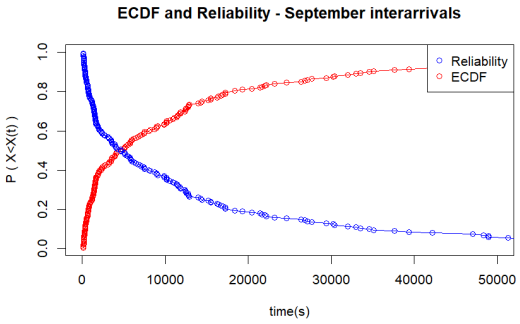


(a) ECDF - September

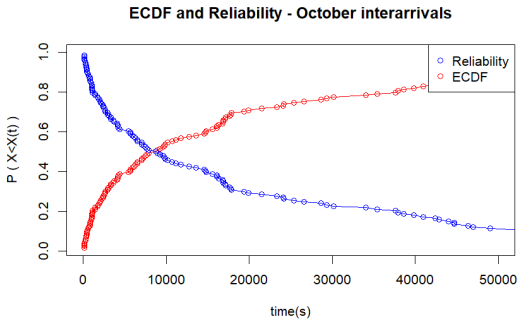


(b) ECDF - October

Se l'obiettivo è quello di avere la probabilità che il sistema sia ancora attivo dopo un certo numero di secondi basta considerare l'evento complementare a quello rappresentato dalla ECDF. Possiamo ottenere una rappresentazione grafica generando un plot della funzione così definita: `r<-1-TTF(t)`.



(a) ECDF and Reliability - September

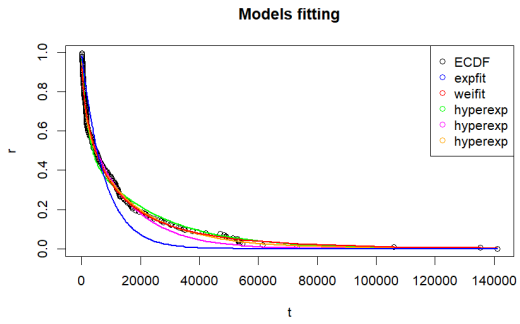


(b) ECDF and Reliability - October

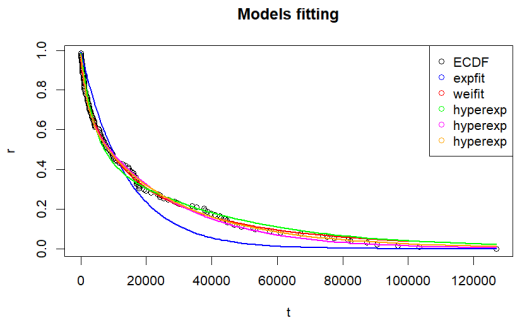
Per meglio predirre l'andamento futuro delle failure basandoci sui dati disponibili è possibile definire un modello di regressione basato sui dati della reliability osservata.
R ci permette di fare regressione non lineare tramite la funzione nls. Questa funzione accetta una formula come modello da adattare ai dati e dei valori di partenza. Nel nostro caso sono stati definiti un modello esponenziale, un modello weibull e tre variazioni di iperesponenziale, come segue.

```
expfit <- nls(r ~ exp(-(l * t)), start = list(l = 1/mean(interarrivals$V1)))
weifit <- nls(r ~ exp(-(l * t)^a), start = list(l = 1/mean(interarrivals$V1), a = 0.9))

hyperexp1 <- nls(r ~ 0.5 * exp(-(l1 * t)) + 0.5 * exp(-(l2 * t)), start = list(
  l1 = 1/mean(interarrivals$V1),
  l2 = 0.1/mean(interarrivals$V1)))

hyperexp2 <- nls(r ~ 0.3 * exp(-(l1 * t)) + 0.7 * exp(-(l2 * t)), start = list(
  l1 = 1/mean(interarrivals$V1),
  l2 = 0.1/mean(interarrivals$V1)))

hyperexp3 <- nls(r ~ 0.5 * exp(-(l1 * t)) + 0.3 * exp(-(l2 * t)) + 0.2 * exp(-(l3 * t)),
start = list(
  l1 = 0.1/mean(interarrivals$V1),
  l2 = 1/mean(interarrivals$V1),
  l3 = 0.9/mean(interarrivals$V1)
))
```

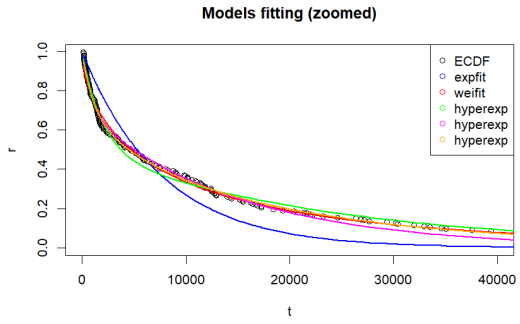
Per avere un riscontro visivo sui modelli generati è utile creare uun grafico che sovrappone l'andamento della reliability empirica a quello dei modelli.



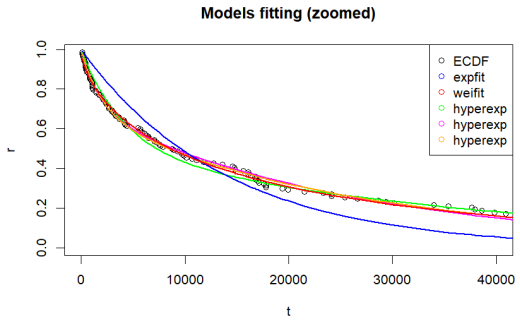
(a) Models fitting - September



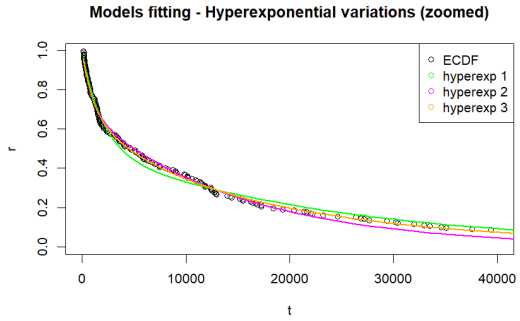
(b) Models fitting - October



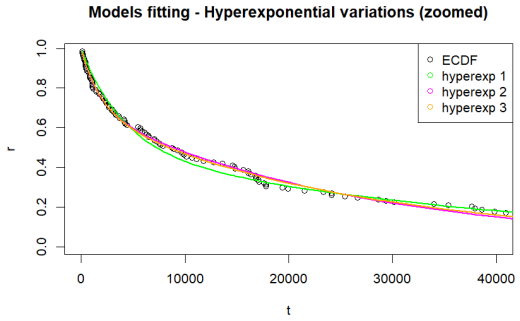
(c) Models fitting zoomed - September



(d) Models fitting zoomed - October



(e) Hyperexponential variations - September



(f) Hyperexponential variations - October

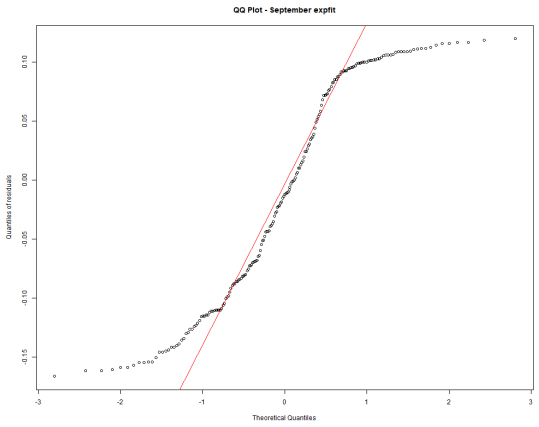
Dai grafici risulta evidente che i modelli di tipo iperesponenziale approssimino meglio l'andamento della reliability empirica e rappresentano la presenza di più fenomeni esponenziali che si sommano all'interno del fenomeno generale osservato.

Per valutare correttamente la bontà di un modello bisogna andare oltre l'analisi visiva ed effettuare il test di Kolmogorov-Smirnov. Questo test è utilizzato per valutare la probabilità che due set di osservazioni provengano dalla stessa distribuzione (p-value). In particolare utilizzando la funzione ks.test è possibile effettuare il test su R ed ottenere il p-value, che ci aspettiamo essere grande ($>> 0.05$) se il modello prodotto è un buon modello. Dal test viene confermato l'iperesponenziale come best performer e in particolare il modello `hyperexp3`.

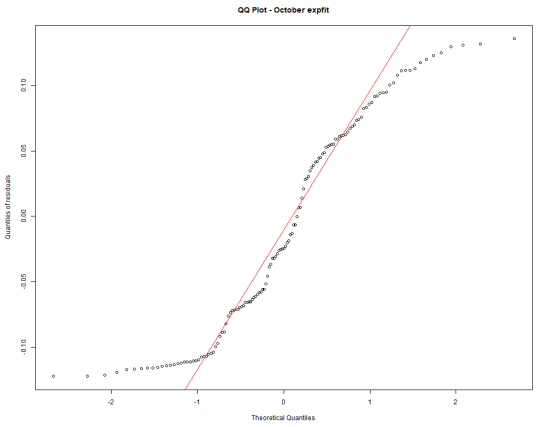
2.4 Analisi aggiuntive

Oltre al test di K-S sono state svolte analisi aggiuntive sulle performance dei modelli. La prima analisi riguarda il grafico quantile-quantile o QQplot.
Il grafico quantile-quantile confronta due distribuzioni e dal esso si può capire quanto siano simili, osserando l'allineamento dei punti rispetto ad una retta diagonale calcolata a partire dai valori del primo e del terzo quartile. R ci permette di effettuare un grafico QQ con la funzione `qqnorm()` e di generare la diagonale con `qqline()`.
Nel caso dei nostri dati l'interesse è quello di sapere se i residui, ovvero la differenza tra valore predetto e valore osservato, seguano una distribuzione normale, poiché ciò è indice di un buon modello. Per fare questo confrontiamo i residui ottenuti con la funzione `tresiduals()` con una

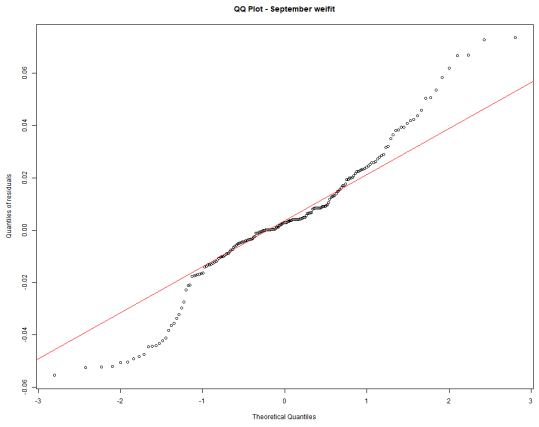
distribuzione normale e generiamo il grafico quantile-quantile. Questo processo è stato ripetuto per tutti e cinque i modelli considerati.



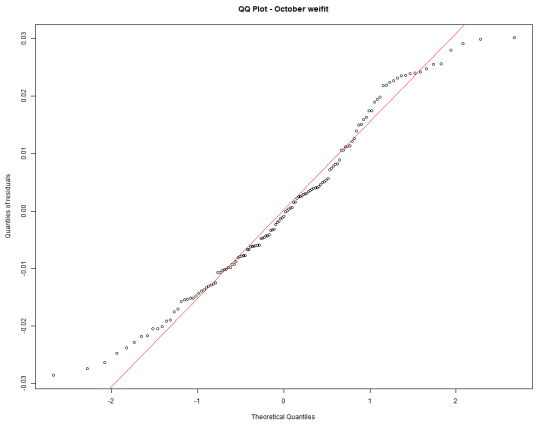
(a) QQ Plot - September expfit



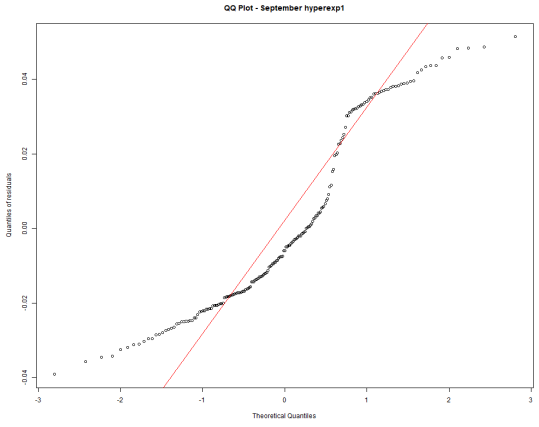
(b) QQ Plot - October expfit



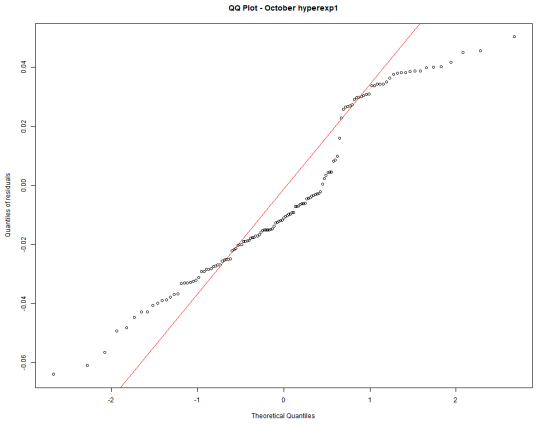
(a) QQ Plot - September weifit



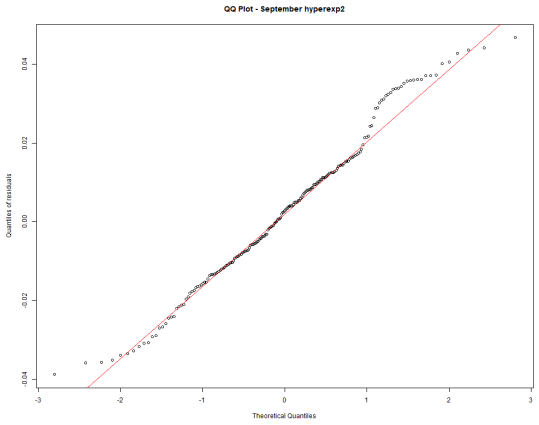
(b) QQ Plot - October weifit



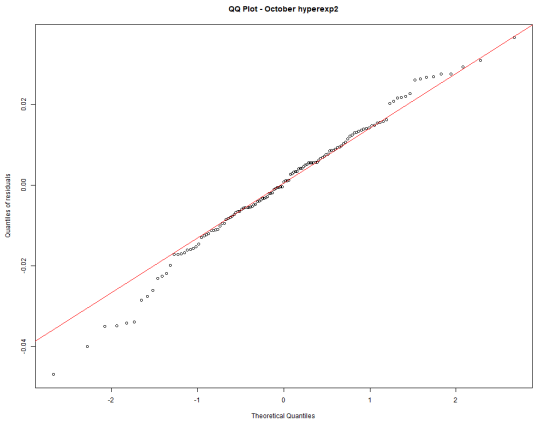
(a) QQ Plot - September hyperexp1



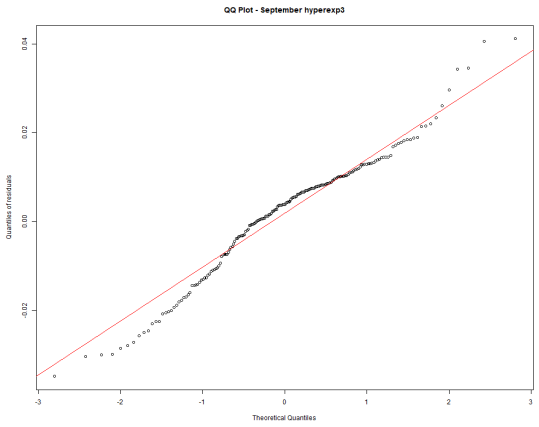
(b) QQ Plot - October hyperexp1



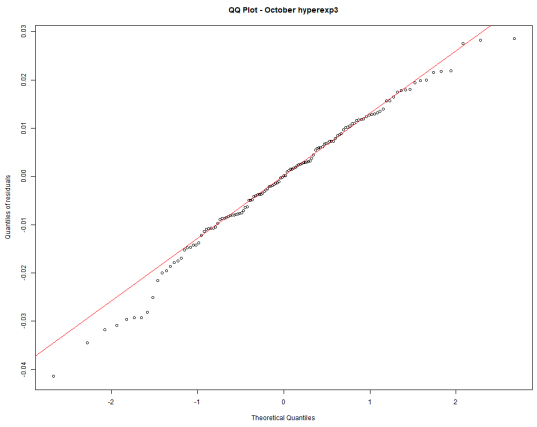
(a) QQ Plot - September hyperexp2



(b) QQ Plot - October hyperexp2



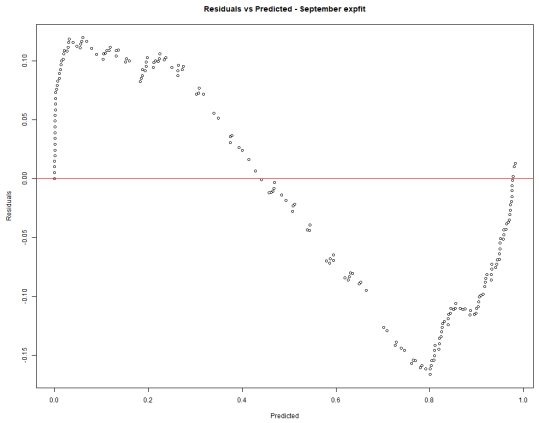
(a) QQ Plot - September hyperexp3



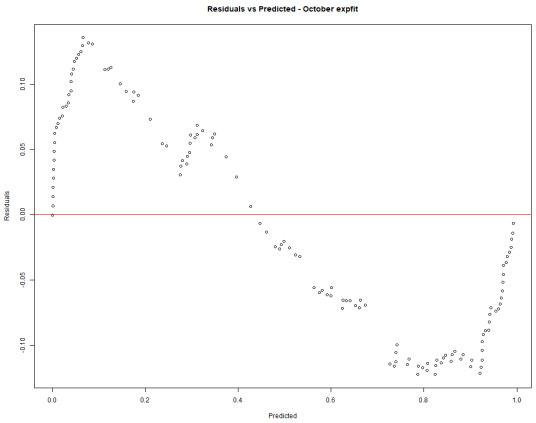
(b) QQ Plot - October hyperexp3

Dal grafico si evince che i residui dei modelli generati tramite esponenziale e weibull non sono aderenti ad una distribuzione normale, così come quelli per la prima variazione di iperesponenziale. Un secondo aspetto che si può evincere è come il tuning dei parametri dell'iperesponenziale abbia portato ad un netto miglioramento nella distribuzione dei residui e quindi nell'accuratezza della predizione.
Osservando con particolare attenzione i grafici ottenuti con le funzioni expfit e weibull, si può notare come si riconducano alla tipologia di qqplot short-tail, ad indicare che con queste funzioni, rispetto ad una distribuzione normale, ci sono più dati collocati agli estremi.
Le stesse valutazioni fatte per il mese di Settembre sono valide per il mese di Ottobre, in cui il il miglior modello sembra essere la terza variazione di iperesponenziale.

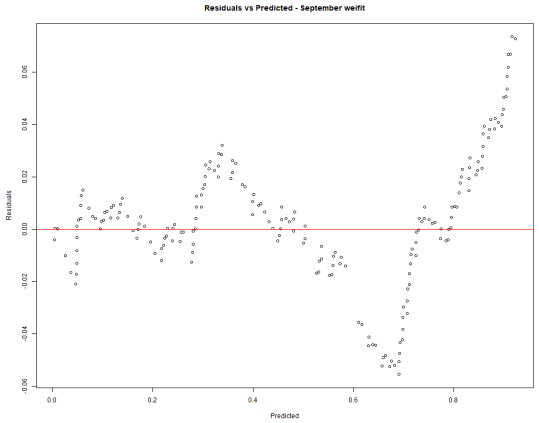
In seguito a questo test sono stati prodotti gli scatterplot relativi a risposta predetta vs residui. L'obiettivo in questo caso è individuare dei trend nello scatterplot per capire se i residui hanno qualche dipendenza dalle variabili predittori.



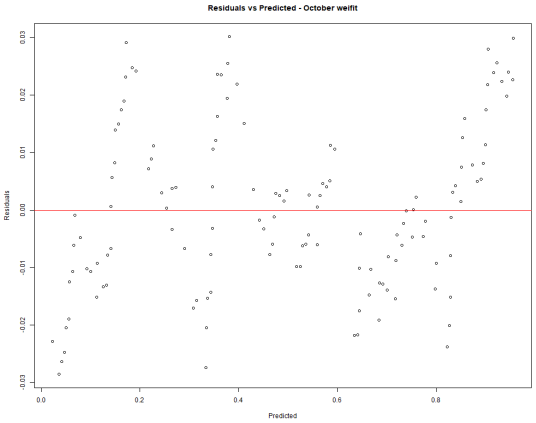
(a) Residuals vs Predicted - September expfit



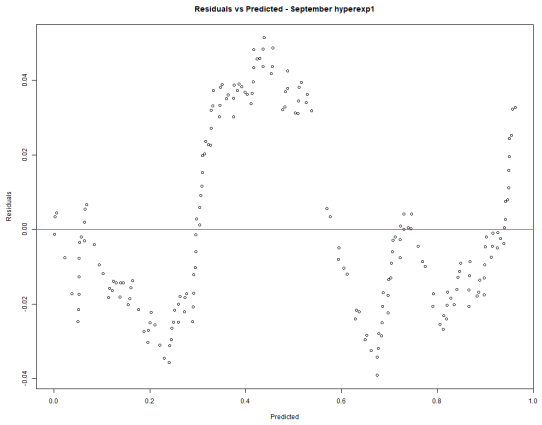
(b) Residuals vs Predicted - October expfit



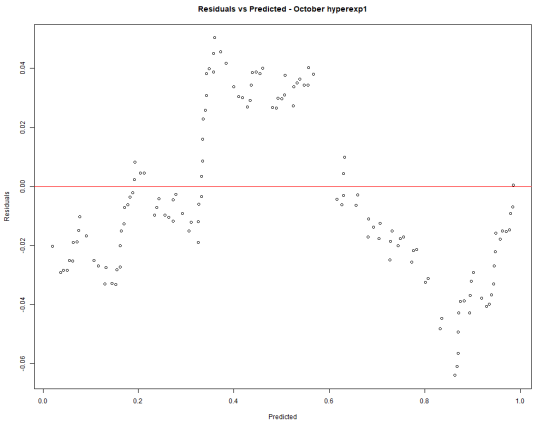
(a) Residuals vs Predicted - September weifit



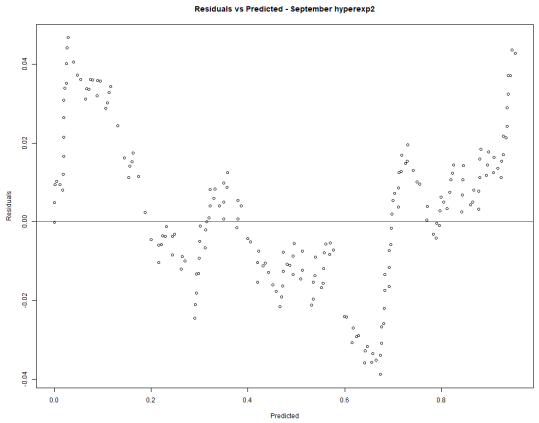
(b) Residuals vs Predicted - October weifit



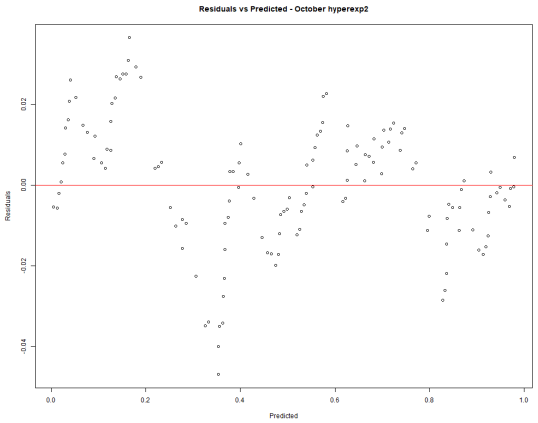
(a) Residuals vs Predicted - September hyperexp1



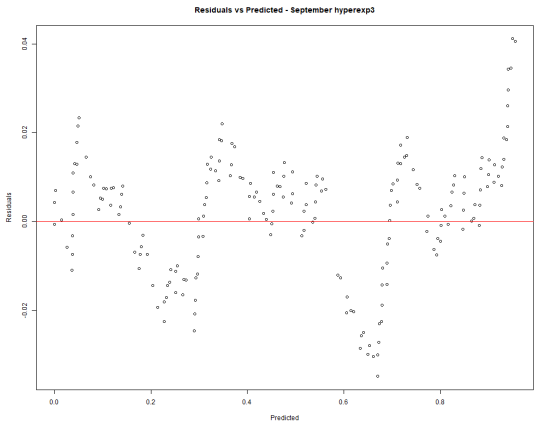
(b) Residuals vs Predicted - October hyperexp1



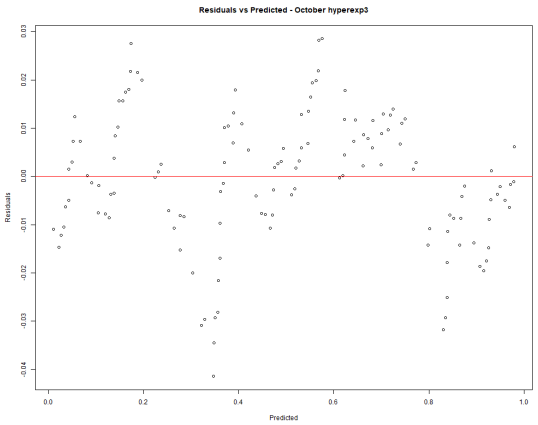
(a) Residuals vs Predicted - September hyperexp2



(b) Residuals vs Predicted - October hyperexp2

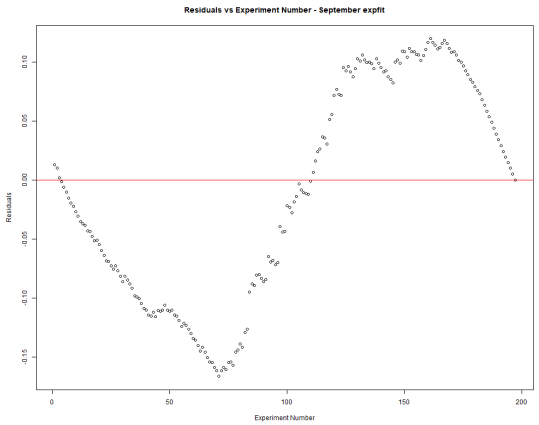


(a) Residuals vs Predicted - September hyperexp3

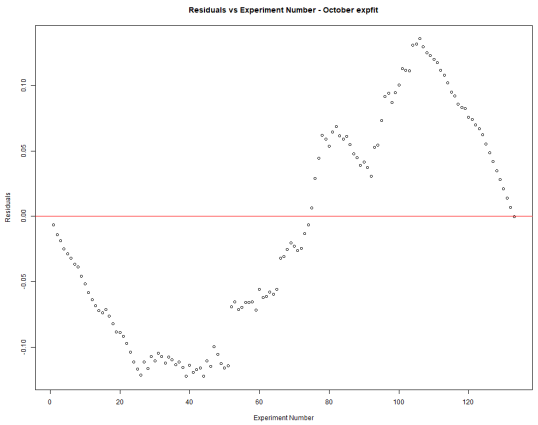


(b) Residuals vs Predicted - October hyperexp3

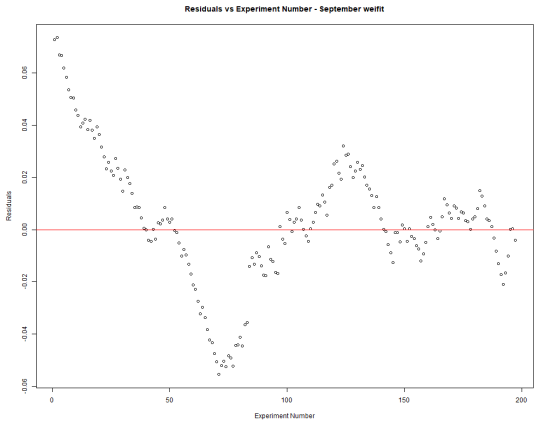
Da questi grafici è evidente che esistono dei trend nell'andamento dei residui rispetto alla risposta predetta, ma è anche importante notare che questi trend sono meno presenti nel modello iperesponenziale. Ciò potrebbe indicare che il modello iperesponenziale cattura meglio la struttura sottostante dei fenomeni nei dati rispetto agli altri modelli esaminati.
L'ultimo tipo di grafico prodotto è quello dei residui sui dati osservati. Questo grafico, come il precedente, ci permette di capire se ci sono trend nei residui e di conseguenza di capire se ci sono stati errori di bias nel metodo in cui sono state effettuate le misuraizoni o se ci sono dei fenomeni non considerati che influenzano i dati.



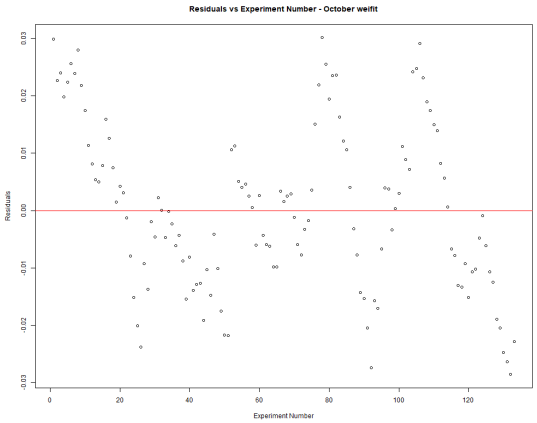
(a) Residuals vs Experiment Number - September expfit



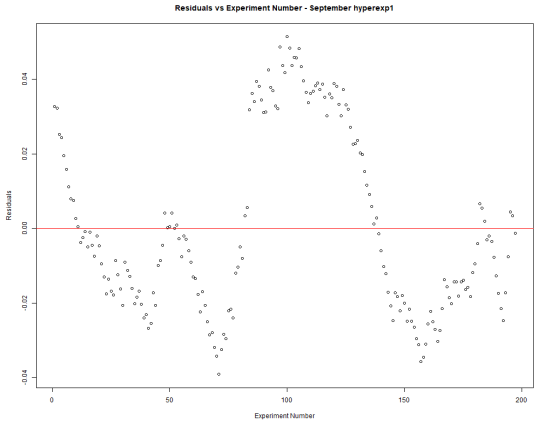
(b) Residuals vs Experiment Number - October expfit



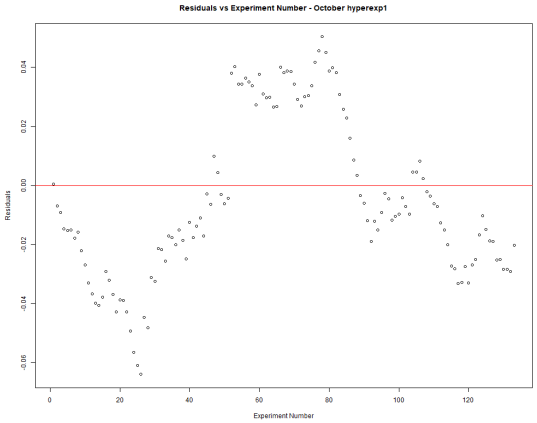
(a) Residuals vs Experiment Number - September weifit



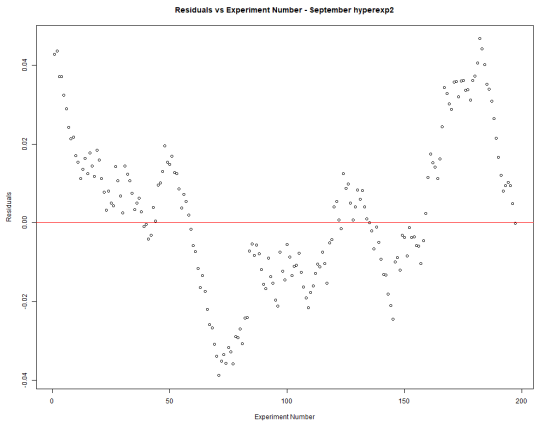
(b) Residuals vs Experiment Number - October weifit



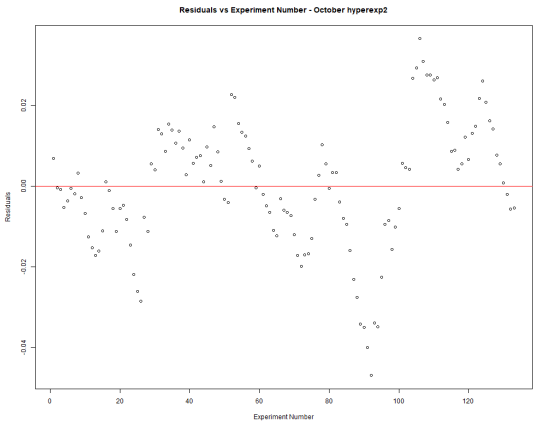
(a) Residuals vs Experiment Number - September hyperexp1



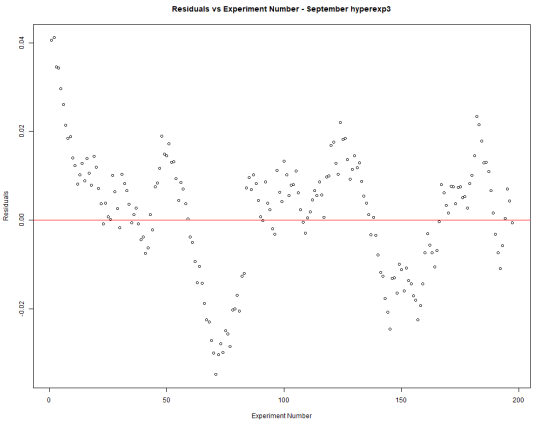
(b) Residuals vs Experiment Number - October hyperexp1



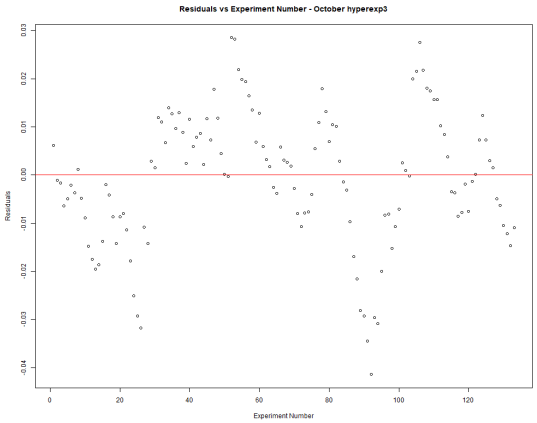
(a) Residuals vs Experiment Number - September hyperexp2



(b) Residuals vs Experiment Number - October hyperexp2



(a) Residuals vs Experiment Number - September hyperexp3



(b) Residuals vs Experiment Number - October hyperexp3

Proprio come per gli scatterplot precedenti esiste un trend tra il valore dei residui e il valore osservato. Questo potrebbe essere dovuto a fenomeni sottostanti a quello osservato che influenzano l'andamento dei dati, ma che non sono correttamente modellati dai modelli prodotti.

2.5 Itemset frequenti

Allo scopo di sapere se ci sono nodi che falliscono spesso insieme è stata effettuata una analisi sui log per generare degli itemset frequenti.
Il primo passo è stato realizzare uno script Python, `transaction_generator.py`, che serve a generare delle transazioni a partire dalle tuple estratte dal file di log. In seguito, tramite uno script R, sono stati estratti gli itemset frequenti a partire dalle transazioni.

```
rulessept<-apriori(datasept, parameter = list(support=0.25, minlen=2,
                                              maxlen=50,
                                              target="frequent itemsets"))
inspect(sort(rulessept, by="support"))


rulesoct<-apriori(dataoct, parameter = list(support=0.25, minlen=2,
                                            maxlen=20,
                                            target="frequent itemsets"))
inspect(sort(rulesoct, by="support"))
```

I risultati di questa operazione variano molto tra i mesi di settembre e ottobre.
Per il mese di settembre sono stati riscontrati itemsets frequenti con nodi appartenenti, principalmente, ai rack 63 - midplane 0 e rack 00 - midplane 0. Questa informazione ci permette di giustificare eventuali collisioni riscontrate, in quanto a fallimento dell'intero rack corrispondono diversi fallimenti a livello di singolo nodo.
Nella Tabella 3 sono visualizzati gli itemsets frequenti ricavati dal mese di settembre con relativo supporto e conteggio.

Per il mese di ottobre sono stati riscontrati poco più di 4000 itemset, tutti con lo stesso supporto di 0,25. Il supporto così basso rende difficile stabilire se questi risultati siano, oppure no, sintomo di collisioni.
Considerando i risultati ottenuti per il mese di settembre, è stato verificato se anche per ottobre gli itemset contengano elementi dello stesso rack o midplane: di 4093 itemset, solo 17 presentano elementi appartenenti allo stesso rack.
Questo risultato risulta abbastanza incoerente rispetto all'andamento riscontrato nel mese di settembre e potrebbe essere un sintomo di qualche evento particolare: considerando la discrepanza tra il numero di failure riscontrati a settembre (81618 raw, 201 tuple) e ottobre (42492 raw, 135 tuple) si potrebbe pensare ad una variazione sostanziale del carico e del tipo di lavoro.
Sarebbero necessarie ulteriori analisi approfondite per identificare oggettivamente l'origine di questa discrepanza tra i mesi di settembre ed ottobre.

Items	Support	Count
{R63-M0-N0, R63-M0-NC}	0.3134328	63
{R63-M0-N4, R63-M0-N8}	0.3084577	62
{R63-M0-N0, R63-M0-N4}	0.3084577	62
{R63-M0-N4, R63-M0-NC}	0.3084577	62
{R63-M0-N0, R63-M0-N8}	0.3084577	62
{R63-M0-N8, R63-M0-NC}	0.3084577	62
{R63-M0-N0, R63-M0-N4, R63-M0-N8}	0.3084577	62
{R63-M0-N4, R63-M0-N8, R63-M0-NC}	0.3084577	62
{R63-M0-N0, R63-M0-N4, R63-M0-NC}	0.3084577	62
{R63-M0-N0, R63-M0-N8, R63-M0-NC}	0.3084577	62
{R63-M0-N0, R63-M0-N4, R63-M0-N8, R63-M0-NC}	0.3084577	62
{R00-M0-N4, R00-M0-N8}	0.2587065	52
{R00-M0-N0, R00-M0-N8}	0.2587065	52
{R00-M0-N8, R00-M0-NC}	0.2587065	52
{R00-M0-N0, R00-M0-N4}	0.2587065	52
{R00-M0-N4, R00-M0-NC}	0.2587065	52
{R00-M0-N0, R00-M0-NC}	0.2587065	52
{R00-M0-N0, R00-M0-N4, R00-M0-N8}	0.2587065	52
{R00-M0-N4, R00-M0-N8, R00-M0-NC}	0.2587065	52
{R00-M0-N0, R00-M0-N8, R00-M0-NC}	0.2587065	52
{R00-M0-N0, R00-M0-N4, R00-M0-NC}	0.2587065	52
{R00-M0-N0, R00-M0-N4, R00-M0-N8, R00-M0-NC}	0.2587065	52
{R06-M0-N8, R06-M0-NC}	0.2537313	51
{R01-M0-N8, R01-M0-NC}	0.2537313	51
{R20-M1-NC, R34-M0-N8}	0.2537313	51
{R20-M1-NC, R34-M0-NC}	0.2537313	51
{R34-M0-N8, R34-M0-NC}	0.2537313	51
{R20-M1-NC, R34-M0-N8, R34-M0-NC}	0.2537313	51

Tabella 3: Frequent itemsets with their support and count - September

3 Analisi sui singoli nodi

In questa sezione è descritta l'esecuzione della seconda parte dell'Homework, ovvero l'analisi condotta sui singoli nodi del sistema. Sarà pertanto descritto il processo adottato per selezionare solo i nodi di maggiore interesse e, per ognuno di questi, il processo di analisi effettuato e i relativi risultati.

3.1 Estrazione dei nodi maggiormente rilevanti

Per poter analizzare tutti i record facenti parti sia del mese di settembre che nel mese di ottobre, è stato innanzitutto necessario unire i due files precedenemente analizzati (ovvero i files **bgloct_1** e **bglsep_1**) in un unico file: **bgl_1**.

Per fare ciò è bastato eseguire delle semplici istruzioni bash:

```
cat bgloct_1 > bgl_1
cat bglsep_1 >> bgl_1
```

Una volta ottenuto il file bgl_1, su di questo è stato eseguito lo script python **statistics.py** per ottenere la lista dei 20 nodi che appaiono più frequentemente nelle righe contenute nel file.

L'esecuzione dello script è stata così effettuata:

```
python statistics.py bgl_1
```

L'output ottenuto è il seguente:

NODE	ENTRIES
R63-M1-NC	497
R62-M0-N4	473
R63-M1-N0	434
R63-M1-N8	422
R62-M0-NC	370
R62-M0-N0	368
R62-M0-N8	365
R64-M1-N0	363
R62-M1-NC	361
R61-M1-N4	360
R63-M1-N4	360
R62-M1-N0	356
R62-M1-N4	354
R62-M1-N8	354
R61-M0-N8	351
R65-M0-N0	346
R61-M0-NC	336
R45-M0-N8	332
R51-M0-N8	330
R60-M1-NC	328

Da questo possiamo notare come i 6 nodi di maggiore interresse per le nostre analisi sono:

- R63-M1-NC con 497 entries
- R62-M0-N4 con 473 entries
- R63-M1-N0 con 434 entries
- R63-M1-N8 con 422 entries
- R62-M0-NC con 370 entries
- R62-M0-N0 con 368 entries

3.2 Estrazione dei log relativi ad un singolo nodo

In seguito all'individuazione dei nodi maggiormente significativi, è stato sviluppato un semplice script python per poter isolare le loggate appartenenti ad uno specifico nodo a partire dal file blg_1. Questo script è **select.py**.

L'utilizzo di select.py è il seguente:

```
python select.py -f <file da cui estrarre le loggate>
-n <nodo da ricercare nel file> -o <file in cui salvare l'output>
```

Per estrarre le loggate relative al nodo R63-M1-NC, lo script è stato usato in questo modo:

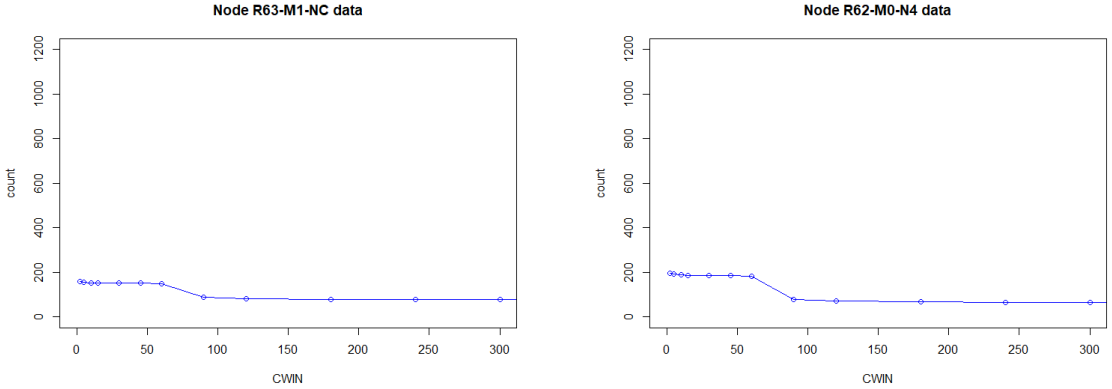
```
python select.py -f bgl_1 -n R63-M1-NC -o extraction_R63-M1-NC.txt
```

In questo modo si è ottenuto il file extraction_R63-M1-NC.txt contenente solo le loggarte relative al nodo R63-M1-NC.

Questo procedimento è stato ripetuto per ognuno dei 6 nodi individuati in precedenza e sono stati creati i rispettivi file in output.

3.3 Calcolo della CWIN

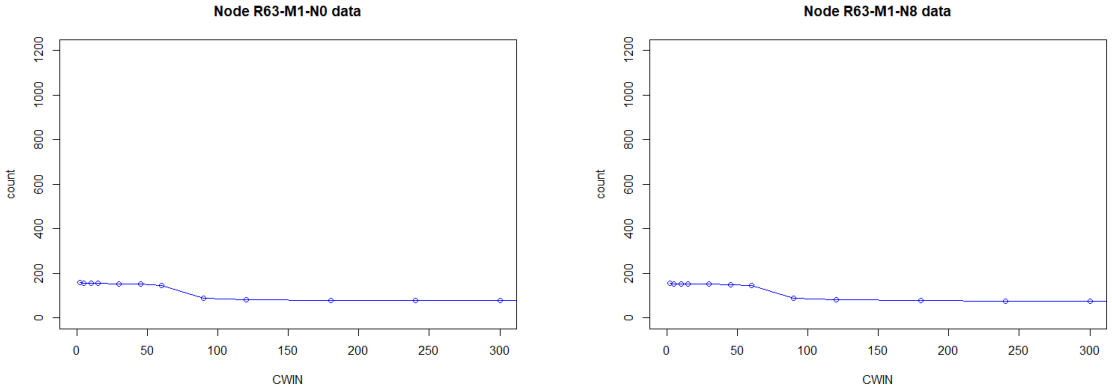
Come mostrato in precedenza per i files relativi ai logs di settembre e ottobre, per ognuno dei nodi individuati precedentemente è stato effettuato il medesimo procedimento, a partire dall'analisi delle possibili CWIN da utilizzarer utilizzando lo script **cwinAnalysis.py** per ogni nodo. I risultati ottenuti sono i seguenti:



(a) Node R63-M1-NC

(b) Node R62-M0-N4

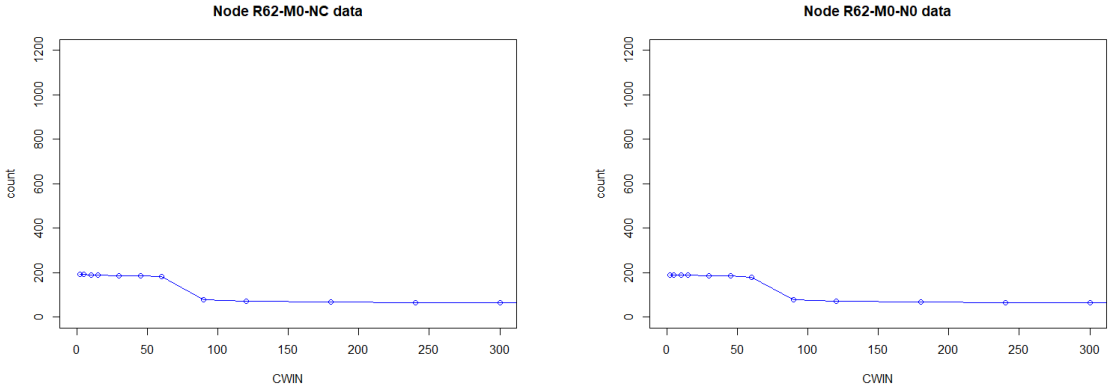
Figura 26: CWIN for nodes



(a) Node R63-M1-N0

(b) Node R63-M1-N8

Figura 27: CWIN for nodes



(a) Node R62-M0-NC

(b) Node R62-M0-N0

Figura 28: CWIN for nodes

3.4 Analisi degli interarrivi

Tramite lo script logCoalescence.py otteniamo i valori degli interarrivi per ogni nodo. Per ottenere una vista generale sulle caratteristiche delle distribuzioni dei valori è stata utilizzata la funzione `summary()` su ognuno dei dataset importati, il cui output è riportato nella tabella sottostante. Si

Node	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
R62-M0-N0	100	1772	41945	69091	93383	591654
R62-M0-N4	100	1728	36404	67274	94640	591654
R62-M0-NC	100	1808	40402	68170	92741	591654
R63-M1-N0	104	1730	20435	59484	86227	591646
R63-M1-N8	105	1448	19959	59487	87907	591646
R63-M1-NC	100	1581	18296	58134	81533	591642

Tabella 4: Summary of the Nodes

può notare come il minimo sia molto vicino o coincidente con la finestra di coalescenza scelta. Tramite valutazioni empiriche è stato osservato, variando la finestra, che questa è una condizione che si verifica sempre poiché i valori degli interarrivi seguono una distribuzione skewed verso sinistra. Allo stesso modo i valori massimi sono molto simili, forse perché dopo quell'intervallo di tempo si verifica un malfunzionamento di natura prevedibile, come ad esempio il riavvio di un nodo di cui questi nodi sono dipendenti per funzionare.

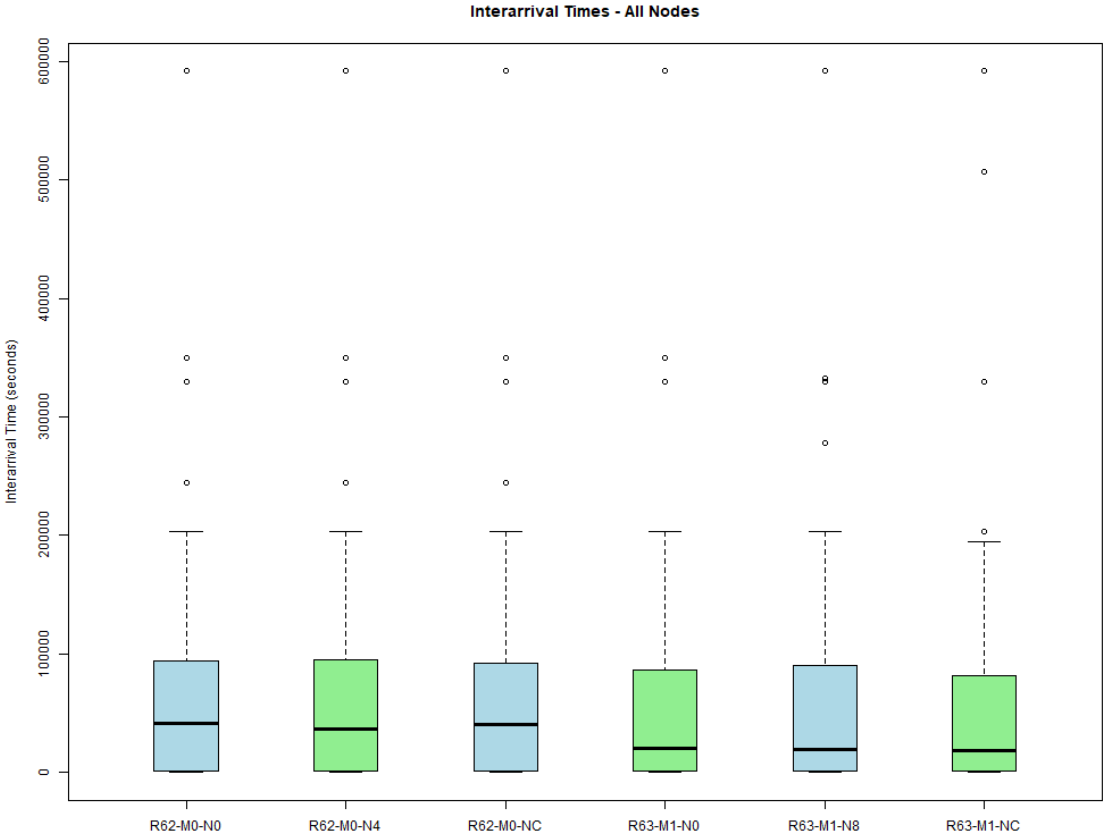
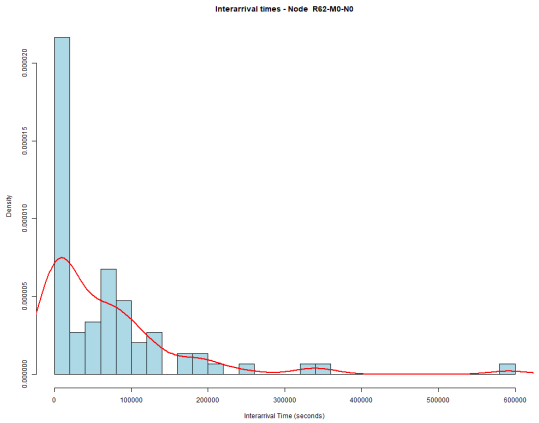
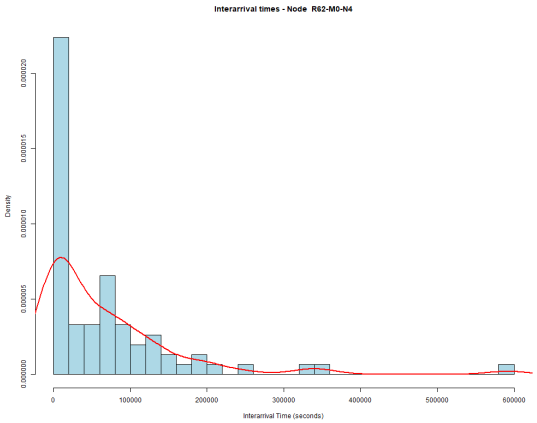


Figura 29: Comparative view of interarrival values distribution for each node

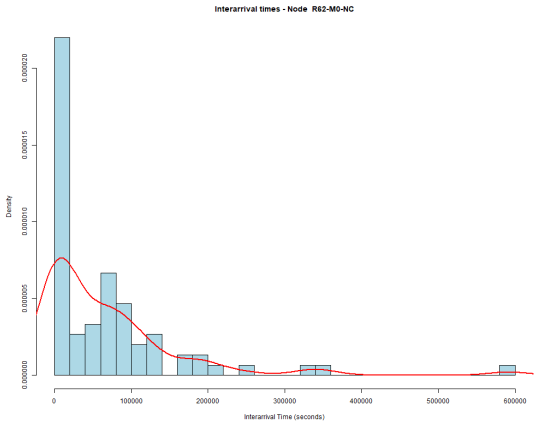
Nel boxplot sono comparate le distribuzioni dei valori per tutti i sei nodi presi in considerazione. E' possibile notare come le distribuzioni siano molto simili, differendo in maniera sostanziale solo per quanto riguarda la mediana dei valori. Si può notare inoltre che i nodi sullo stesso rack hanno una mediana comparabile, indice che sono soggetti allo stesso tipo di failure.



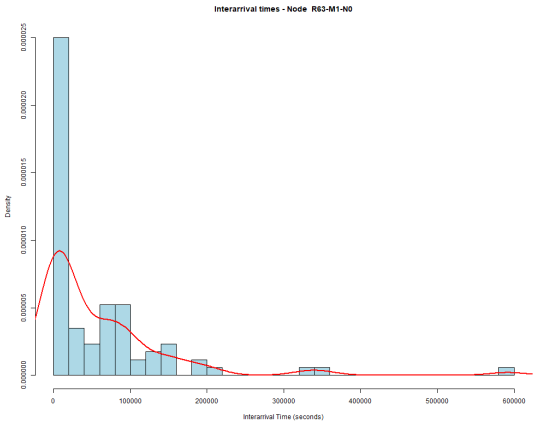
(a) Node R62-M0-N0 - Interarrival times



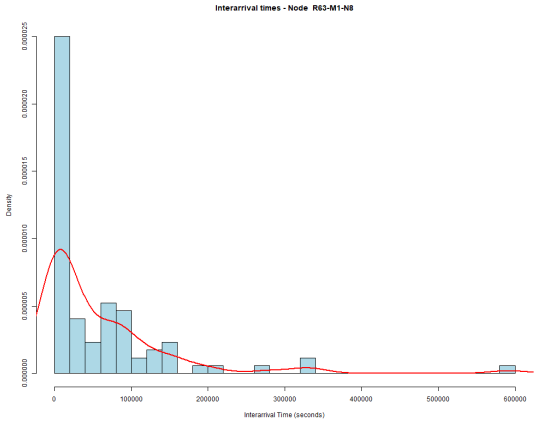
(b) Node R62-M0-N4 - Interarrival times



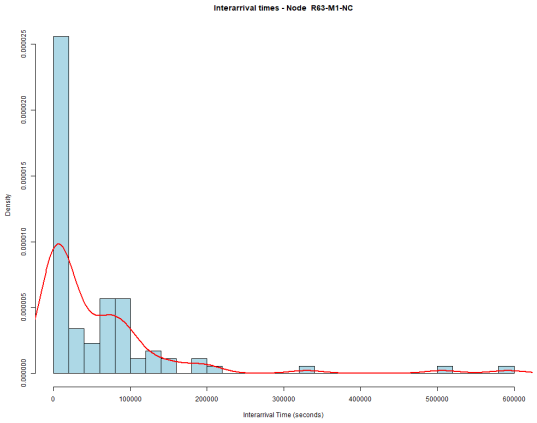
(c) Node R62-M0-NC - Interarrival times



(d) Node R63-M1-N0 - Interarrival times



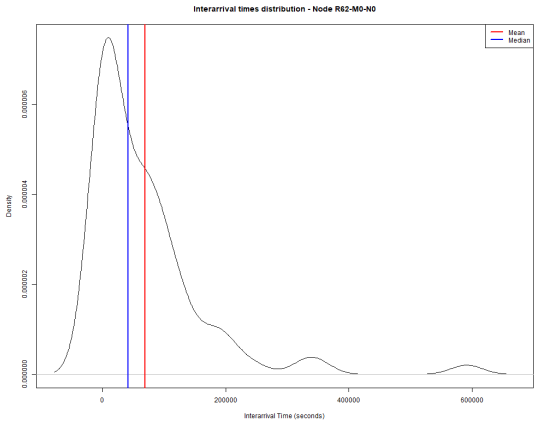
(e) Node R63-M1-N8 - Interarrival times



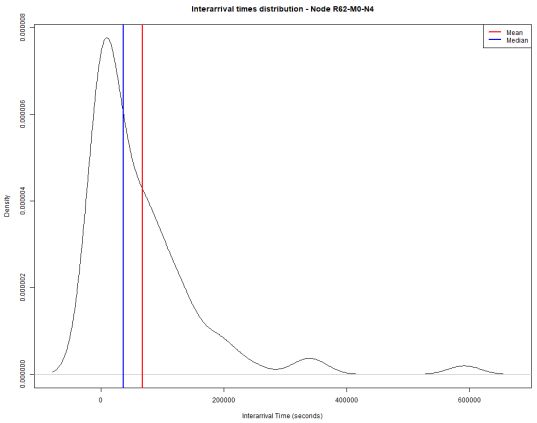
(f) Node R63-M1-NC - Interarrival times

Figura 30: Interarrival times for each node

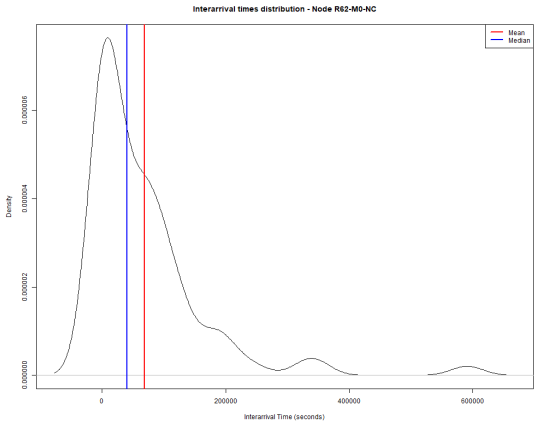
I grafici precedenti evidenziano una informazione che nel boxplot è già presente, ovvero che la distribuzione dei valori degli interarrivi è molto simile per tutti i nodi. Questa è una ulteriore conferma del fatto che le failure possano seguire qualche forma di evento modellabile e predicibile, ma soprattutto che la distribuzione ottenuta nel complesso nei mesi di ottobre e settembre si mantiene anche nei singoli nodi e non è il risultato della sovrapposizione di più distribuzioni dalla forma molto diversa.



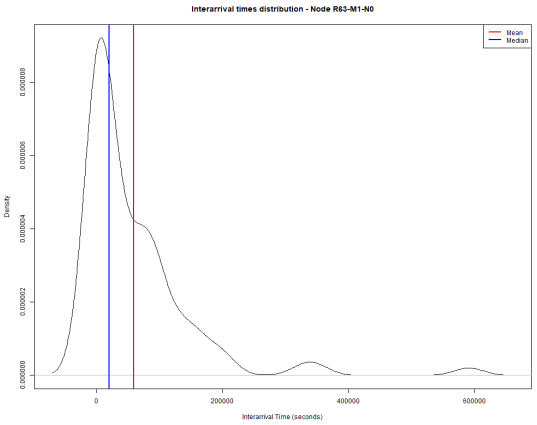
(a) Node R62-M0-N0 - Interarrival times distribution



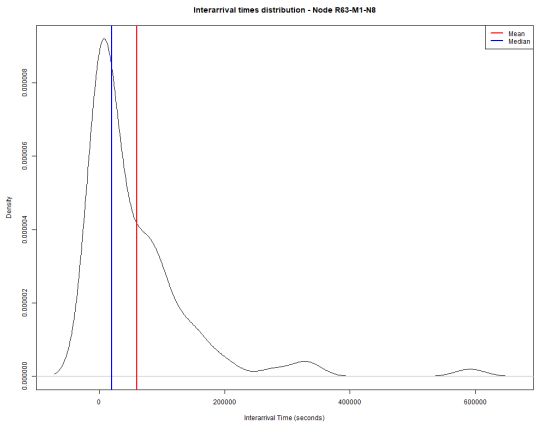
(b) Node R62-M0-N4 - Interarrival times distribution



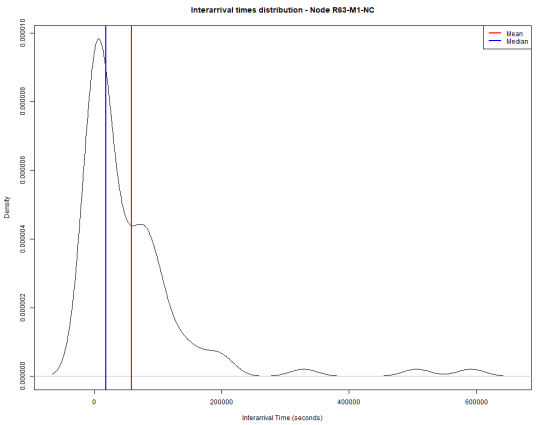
(c) Node R62-M0-NC - Interarrival times distribution



(d) Node R63-M1-N0 - Interarrival times distribution



(e) Node R63-M1-N8 - Interarrival times distribution



(f) Node R63-M1-NC - Interarrival times distribution

Figura 31: Interarrival times distribution for each node

Node	Media	Deviazione standard	CI al 90%	CI al 95%
R62-M0-N0	69091.26	97382.51	[50231.37, 87951.14]	[46529.56, 91652.95]
R62-M0-N4	67274.16	96735.14	[48794.13, 85754.19]	[45169.23, 89379.09]
R62-M0-NC	68170.39	96975.28	[49518.23, 86822.54]	[45858.40, 90482.37]
R63-M1-N0	59484.21	90851.26	[43192.44, 75775.98]	[40005.66, 78962.76]
R63-M1-N8	59486.56	92361.04	[42924.05, 76049.07]	[39684.31, 79288.81]
R63-M1-NC	58134.15	96315.19	[41064.28, 75204.01]	[37726.91, 78541.39]

Tabella 5: Risultati per i vari nodi

Come per le analisi su i mesi di settembre e ottobre, bisogna prendere in considerazione la forma non gaussiana della sitribuzione per ottenere valori più accurati negli intervalli di confidenza. Anche in questo caso è stata utilizzata la procedura di bootstrapping per avere una misura dell'intervallo di confidenza indiretta, effettuata su più sample del dataset di partenza invece che direttamente sui dati.

La procedura di bootstrapping è stata effettuata nel codice grazie all'utilizzo della libreria "boot", e sono stati calcolati gli intervalli di confidenza al 90 e al 95

Node	**Media**	**Deviazione standard**	**CI al 90%**	**CI al 95%**
R62-M0-N0	69091.26	97382.51	[52432, 91808]	[50010, 96151]
R62-M0-N4	67274.16	96735.14	[51093, 89884]	[48977, 95756]
R62-M0-NC	68170.39	96975.28	[54115, 91241]	[51716, 97906]
R63-M1-N0	59484.21	90851.26	[47523, 81391]	[46136, 88456]
R63-M1-N8	59486.56	92361.04	[46225, 79876]	[44360, 86127]
R63-M1-NC	58134.15	96315.19	[44253, 79473]	[42522, 82514]

Tabella 6: Risultati per i vari nodi usando bootstrap

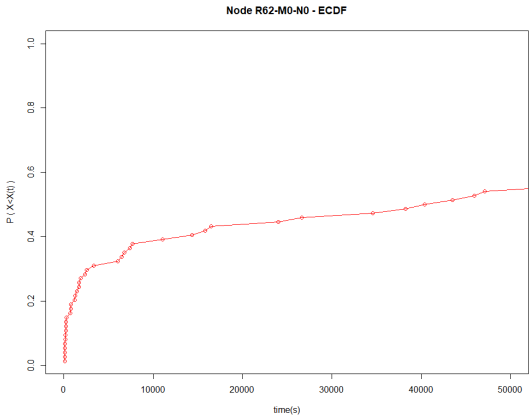
Come ultima analisi di questa sezione, la traccia richiede di effettuare un ranking decrescente dei nodi, basato sulla media del TTF.

Node	**MTTF**
R62-M0-N0	69091.26
R62-M0-NC	68170.39
R62-M0-N4	67274.16
R63-M1-N8	59486.56
R63-M1-N0	59484.21
R63-M1-NC	58134.15

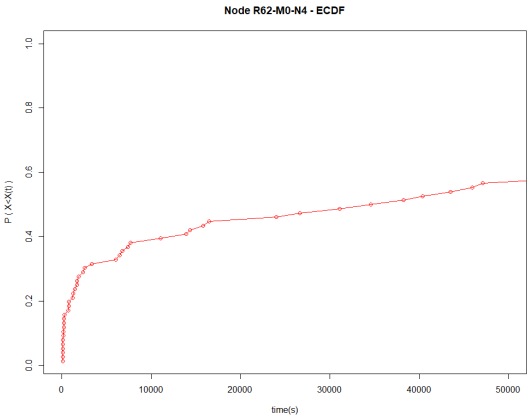
Tabella 7: Ranking decrescente dei nodi, basato su MTTF

3.5 Reliability modelling

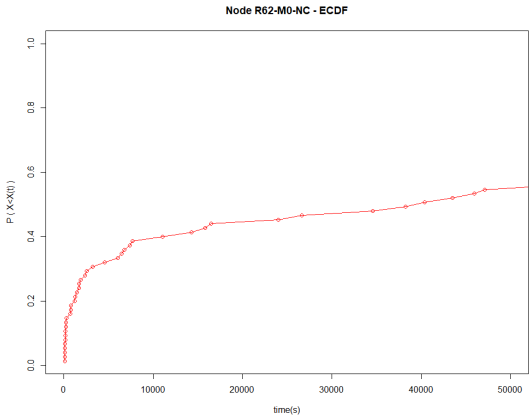
Come fatto in precedenza per i mesi di Settembre e Ottobre, possiamo modellare una CDF empirica partendo dai dati degli interarrivi.



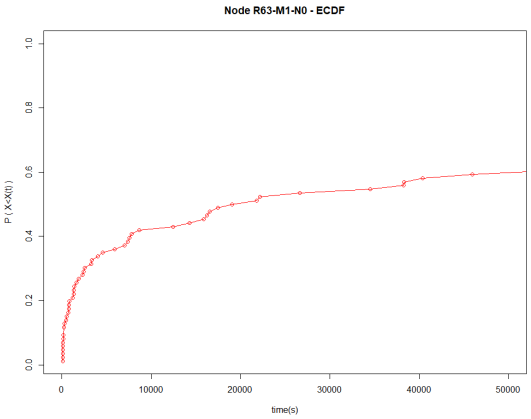
(a) Node R62-M0-N0 - ECDF



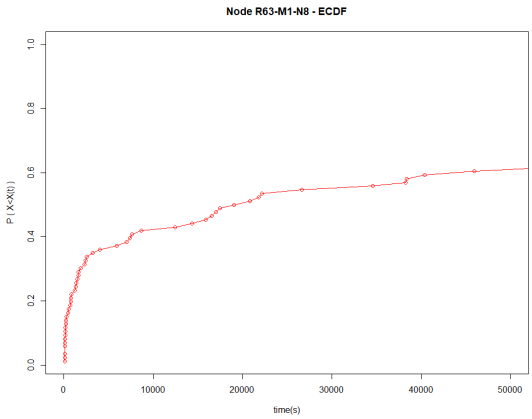
(b) Node R62-M0-N4 - ECDF



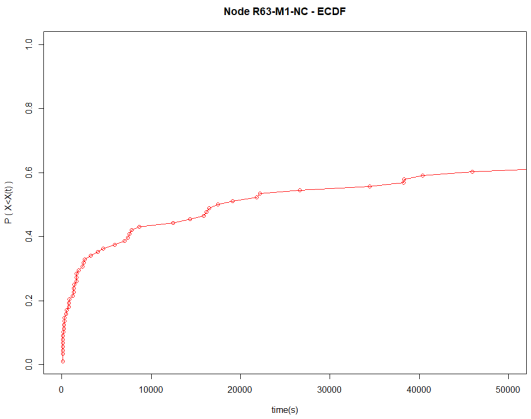
(c) Node R62-M0-NC - ECDF



(d) Node R63-M1-N0 - ECDF



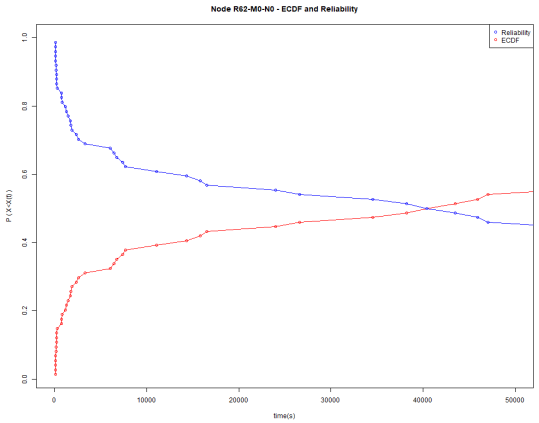
(e) Node R63-M1-N8 - ECDF



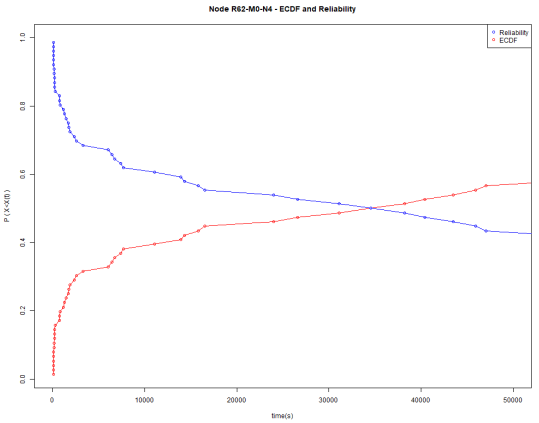
(f) Node R63-M1-NC - ECDF

Figura 32: ECDF for each node

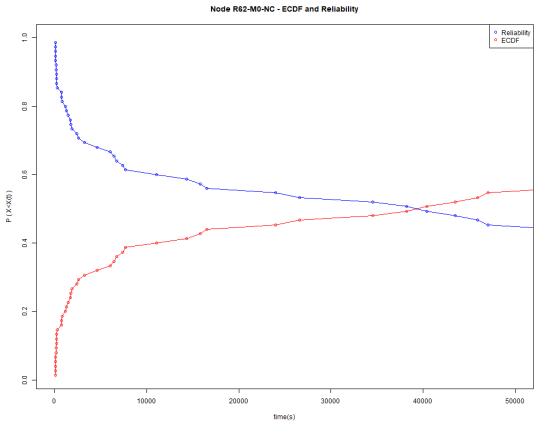
Reiterando quando detto per la prima parte dell'homework, per avere la probabilità che il sistema sia ancora attivo dopo un certo numero di secondi basta considerare l'evento complementare a quello rappresentato dalla ECDF. Possiamo ottenere una rappresentazione grafica generando un plot della funzione così definita: `r<- 1-TTF(t)`.



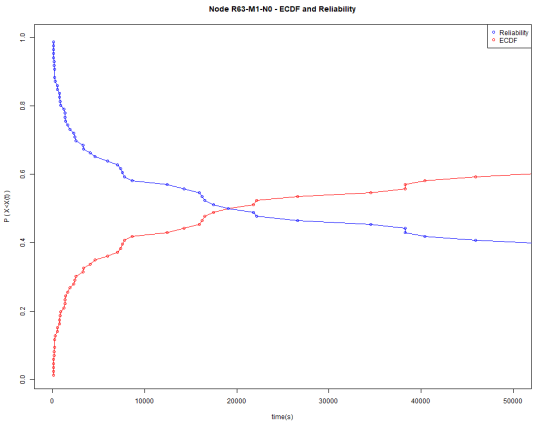
(a) Node R62-M0-N0 - ECDF and Reliability



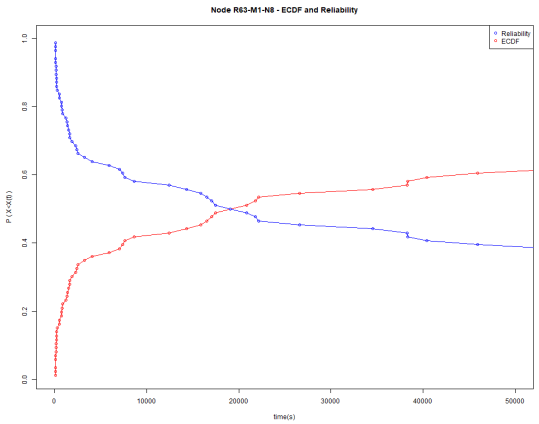
(b) Node R62-M0-N4 - ECDF and Reliability



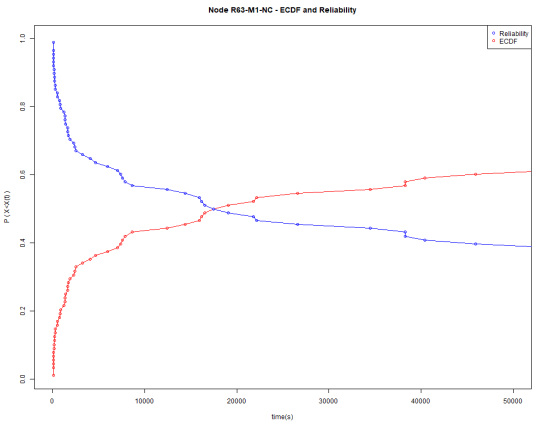
(c) Node R62-M0-NC - ECDF and Reliability



(d) Node R63-M1-N0 - ECDF and Reliability



(e) Node R63-M1-N8 - ECDF and Reliability



(f) Node R63-M1-NC - ECDF and Reliability

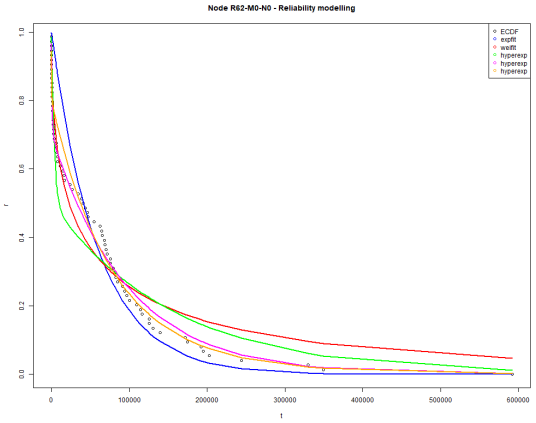
Figura 33: ECDF and Reliability for each node

Per ottenere un modello partendo dai dati della reliability empirica è stata nuovamente utilizzata la funzione nls() di R, con la stessa modalità della parte precedente, ma diversi parametri. In particolare:

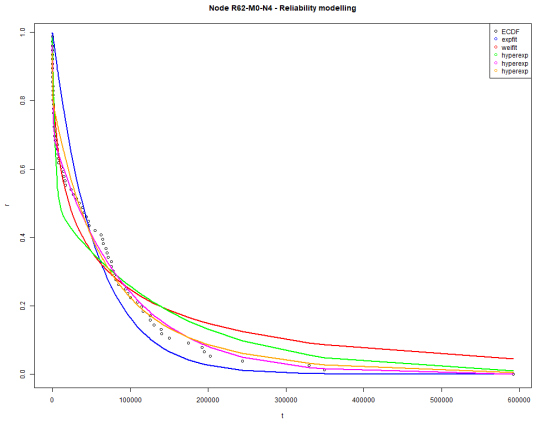
```
    expfit<-nls(r~exp(-(l*t)), start = list(l=1/mean(interarrivals$V1)))
weifit<-nls(r~exp(-(l*t)^a), start = list(l=1/mean(interarrivals$V1), a=0.9))

hyperexp1 <- nls(r~0.5*exp(-(l1*t))+0.5*exp(-(l2*t)), start = list(
  l1=1/mean(interarrivals$V1),
  l2=0.1/mean(interarrivals$V1)
))

hyperexp2 <- nls(r~0.3*exp(-(l1*t))+0.7*exp(-(l2*t)), start = list(
```

```
  l1=1/mean(interarrivals$V1),
  l2=0.1/mean(interarrivals$V1)
))

hyperexp3 <- nls(r ~ 0.2 * exp(-(l1 * t)) + 0.3 * exp(-(l2 * t)) + 0.5 * exp(-(l3 * t)),
                 start = list(
                   l1 = 1 / mean(interarrivals$V1),
                   l2 = 0.5 / mean(interarrivals$V1),
                   l3 = 0.2 / mean(interarrivals$V1)
                 ))
```

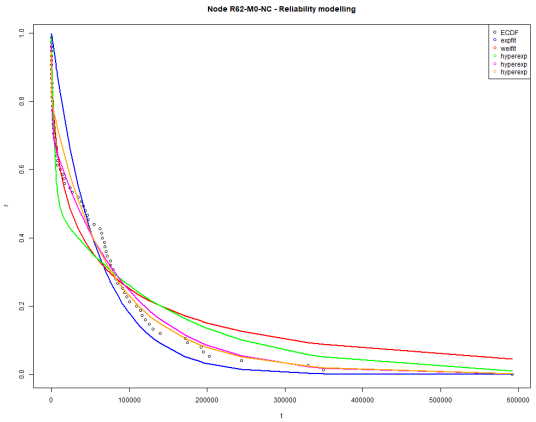
Una volta ottenuti i modelli possiamo compararne l'andamento in maniera visiva, creando un grafico per l'andamento dei vari modelli rispetto alla reliability empirica.



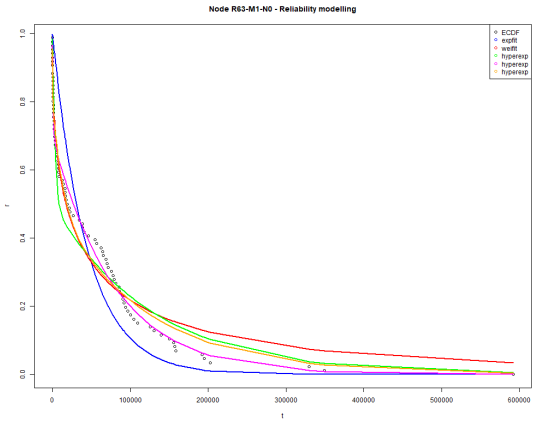
(a) Node R62-M0-N0 - Reliability modelling



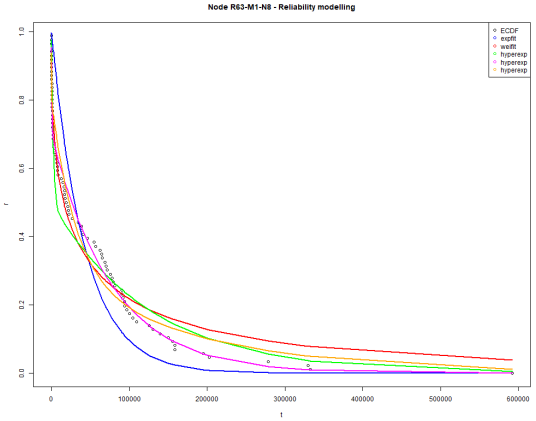
(b) Node R62-M0-N4 - Reliability modelling



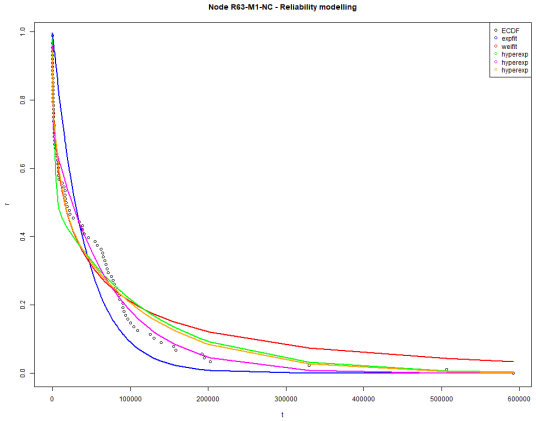
(c) Node R62-M0-NC - Reliability modelling



(d) Node R63-M1-N0 - Reliability modelling



(e) Node R63-M1-N8 - Reliability modelling



(f) Node R63-M1-NC - Reliability modelling

Figura 34: Reliability modelling for each node

Osservando i grafici risulta evidente che i modelli di tipo iperesponenziale approssimino meglio l'andamento della reliability empirica. E' anche possibile osservare come il tuning dei parametri abbia migliorato nettamente le performance dell'iperesponenziale stesso, che altrimenti risulta egualmente poco accurato.

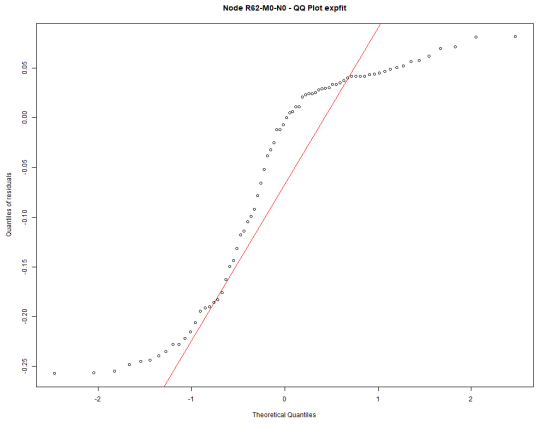
Per valutare correttamente la bontà di un modello bisogna andare oltre l'analisi visiva ed effettuare il test di Kolmogorov-Smirnov. Questo test è utilizzato per valutare la probabilità che due set di osservazioni provengano dalla stessa distribuzione (p-value). In particolare utilizzando la funzione ks.test è possibile effettuare il test su R ed ottenere il p-value, che ci aspettiamo essere grande ($>> 0.05$) se il modello prodotto è un buon modello. Dal test viene confermato l'iperesponenziale come best performer e in particolare il modello `hyperexp2`.

Node	expfit	weifit	hyperexp1	hyperexp2	hyperexp3
R62-M0-N0	0.008718	0.7838	0.5117	0.9964	0.7838
R62-M0-N4	0.01008	0.7973	0.5291	0.997	0.7973
R62-M0-NC	0.009386	0.7907	0.5204	0.9967	0.6562
R63-M1-N0	0.006527	0.8488	0.6016	0.9987	0.9376
R63-M1-N8	0.002166	0.8488	0.4773	0.985	0.8488
R63-M1-NC	0.002542	0.8585	0.6164	0.9868	0.943

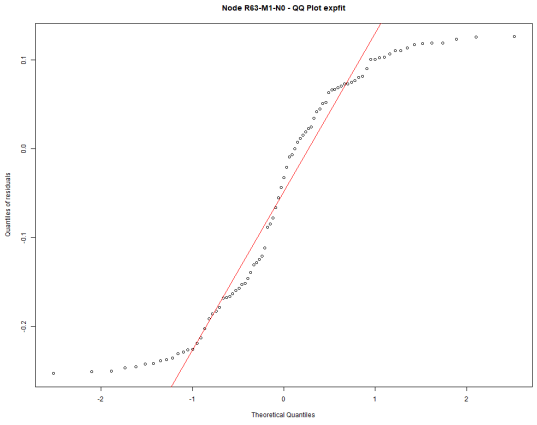
Tabella 8: P-values from the Kolmogorov-Smirnov test for each node and model

3.6 Analisi aggiuntive

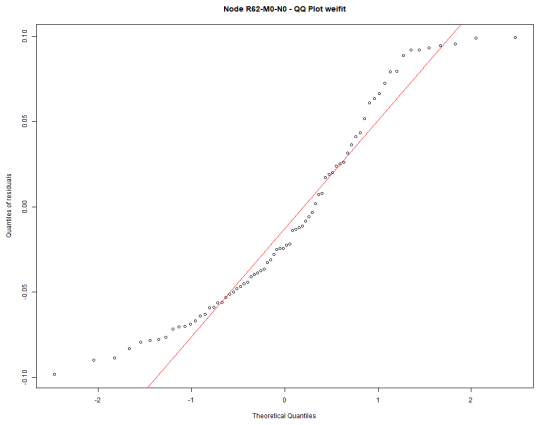
Come per la parte 1, anche per i singoli nodi sono state effettuate analisi riguardo ai grafici quantile-quantile e dei residui. Per brevità sul documento sono riportate solo le immagini relative ad un nodo del rack 63 e uno del rack 62, ma le altre sono visionabili sul progetto allegato al report. Il primo grafico prodotto è il grafico quantile-quantile, per verificare l'ipotesi di normalità della sitribuzione dei residui, per i vari modelli prodotti.



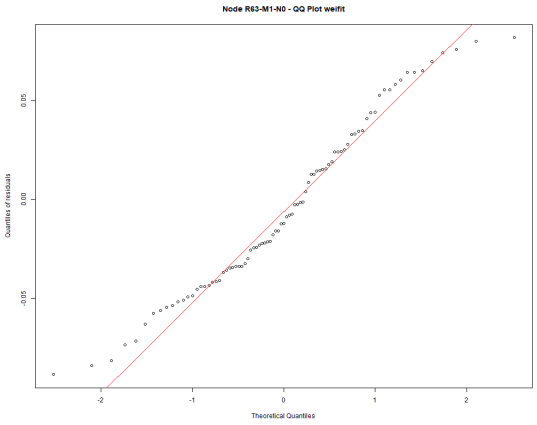
(a) QQ Plot - Node R62-M0-N0 expfit



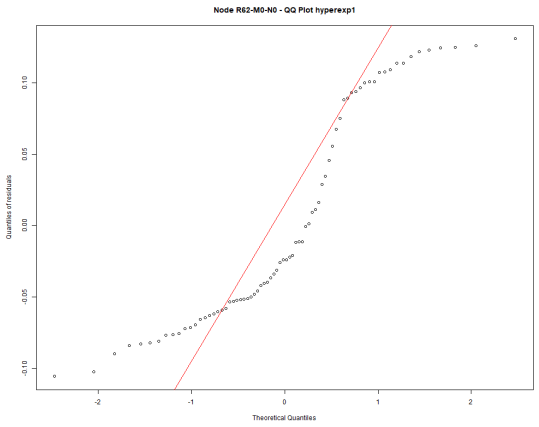
(b) QQ Plot - Node R63-M1-N0 expfit



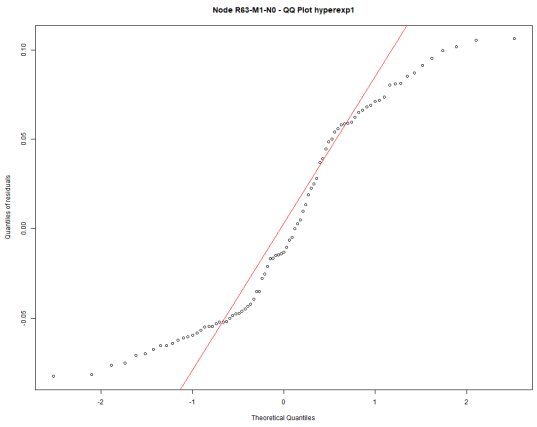
(a) QQ Plot - Node R62-M0-N0 weifit



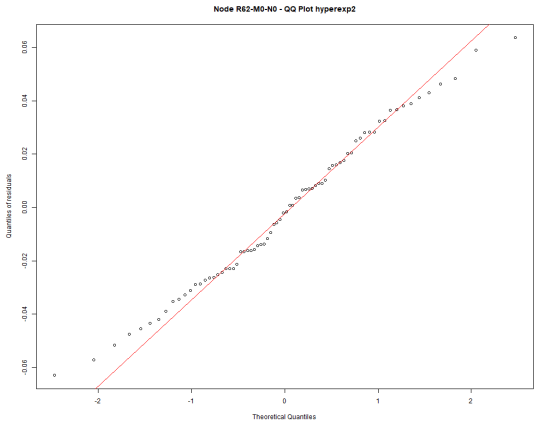
(b) QQ Plot - Node R63-M1-N0 weifit



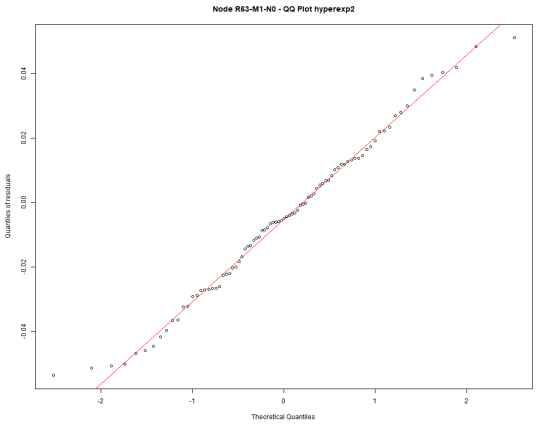
(a) QQ Plot - Node R62-M0-N0 hyperexp1



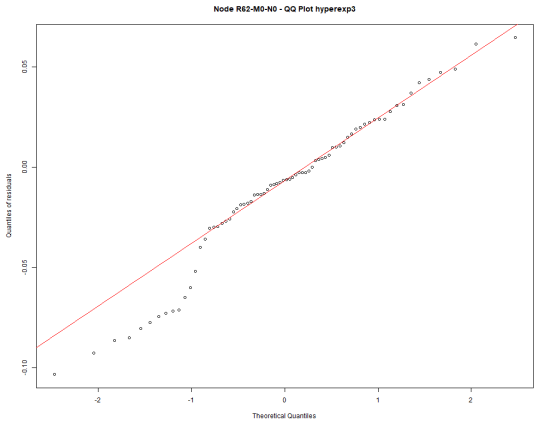
(b) QQ Plot - Node R63-M1-N0 hyperexp1



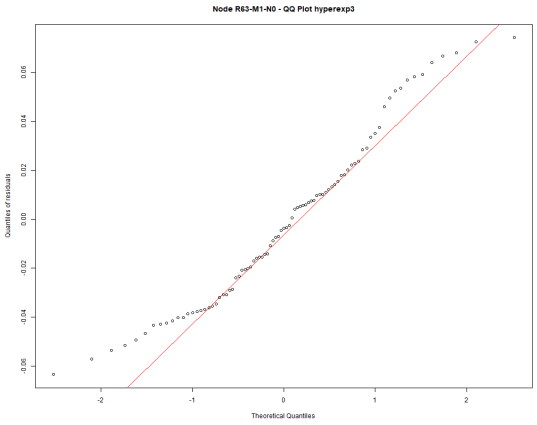
(a) QQ Plot - Node R62-M0-N0 hyperexp2



(b) QQ Plot - Node R63-M1-N0 hyperexp2

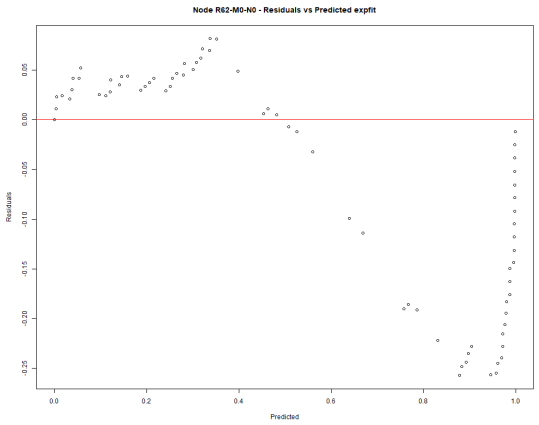


(a) QQ Plot - Node R62-M0-N0 hyperexp3

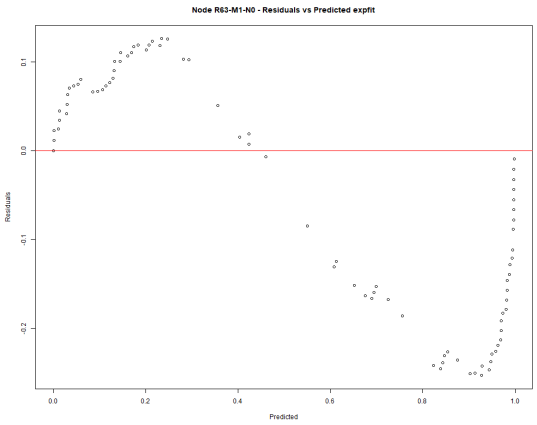


(b) QQ Plot - Node R63-M1-N0 hyperexp3

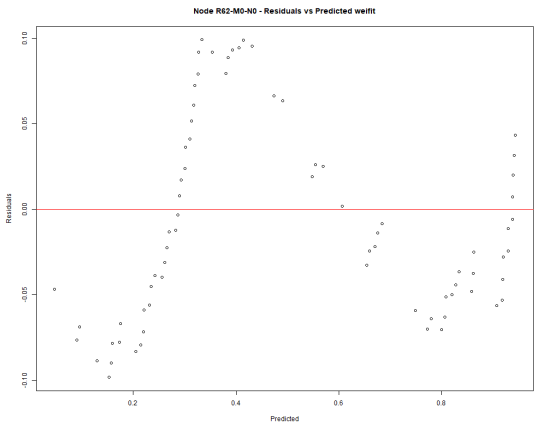
Dai grafici è possibile osservare come la seconda iterazione di iperesponenziale sia effettivamente quasi aderente ad una distribuzione normale dei residui, soprattutto considerando i nodi del rack 63.



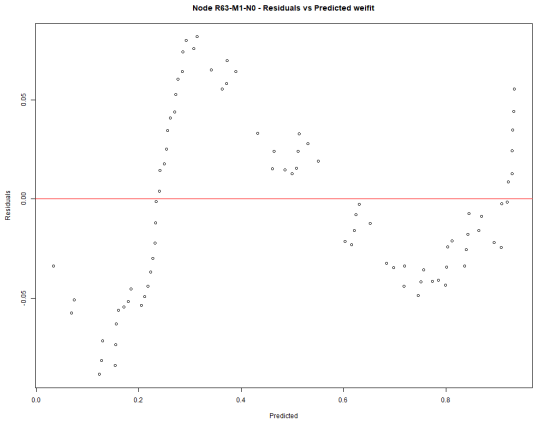
(a) Residuals vs Predicted - Node R62-M0-N0 expfit



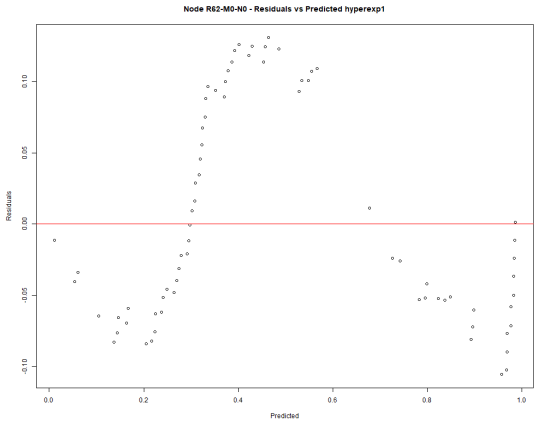
(b) Residuals vs Predicted - Node R63-M1-N0 expfit



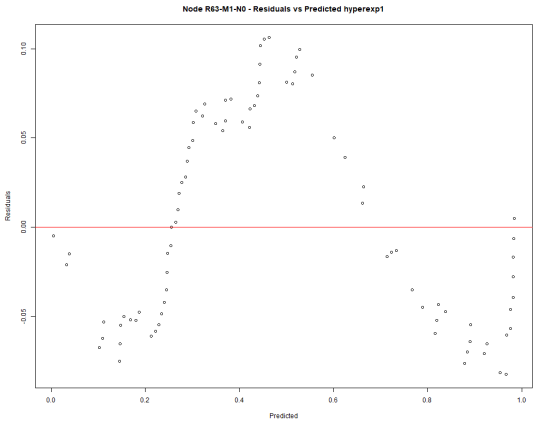
(a) Residuals vs Predicted - Node R62-M0-N0 weifit



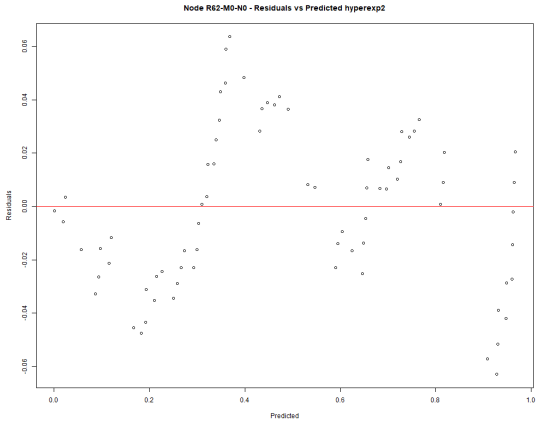
(b) Residuals vs Predicted - Node R63-M1-N0 weifit



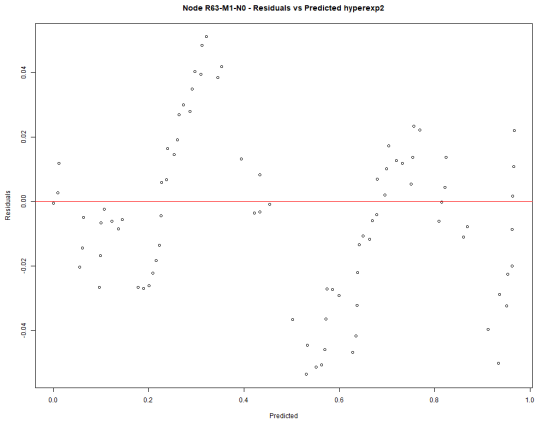
(a) Residuals vs Predicted - Node R62-M0-N0 hyperexp1



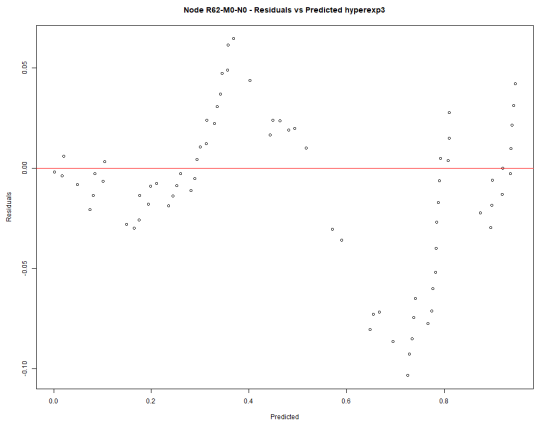
(b) Residuals vs Predicted - Node R63-M1-N0 hyperexp1



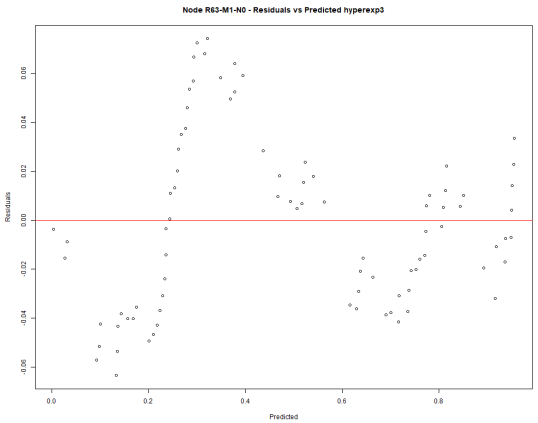
(a) Residuals vs Predicted - Node R62-M0-N0 hyperexp2



(b) Residuals vs Predicted - Node R63-M1-N0 hyperexp2

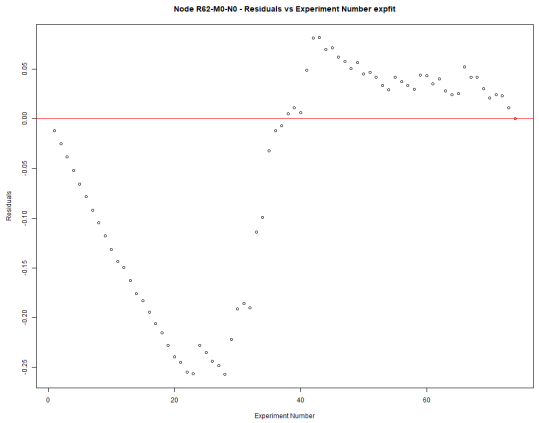


(a) Residuals vs Predicted - Node R62-M0-N0 hyperexp3

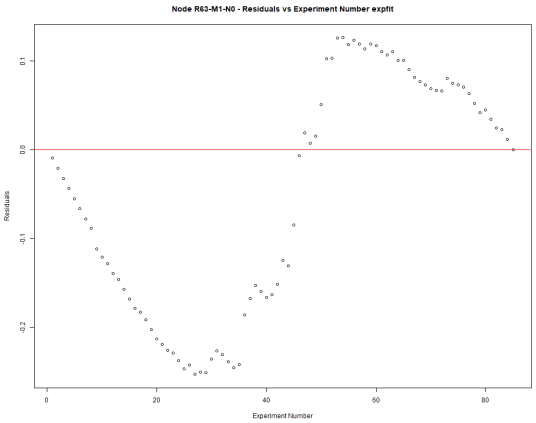


(b) Residuals vs Predicted - Node R63-M1-N0 hyperexp3

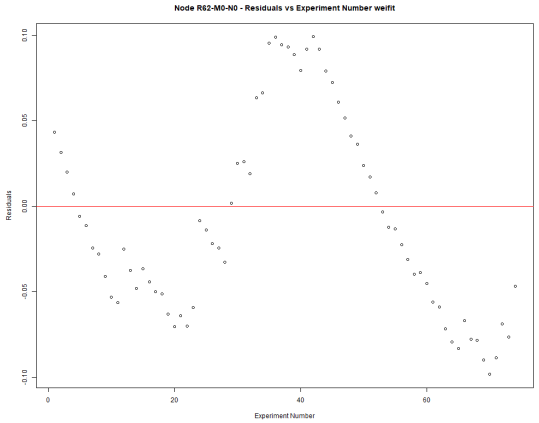
Dai grafici dei residui sulla risposta predetta si può evincere che esistono dei trend nei residui. Questo è indice del fatto che potrebbe esserci una dipendenza tra gli errori e le variabili di predizione e che si dovrebbe variare il modello per tenerne conto. Questa condizione è valida per tutti i nodi analizzati e per tutti i modelli di regressione utilizzati.
Si nota inoltre come per le funzioni expfit il grafico ottenuto sia short-tailed, motivo per cui valgono le considerazioni fatte prima.



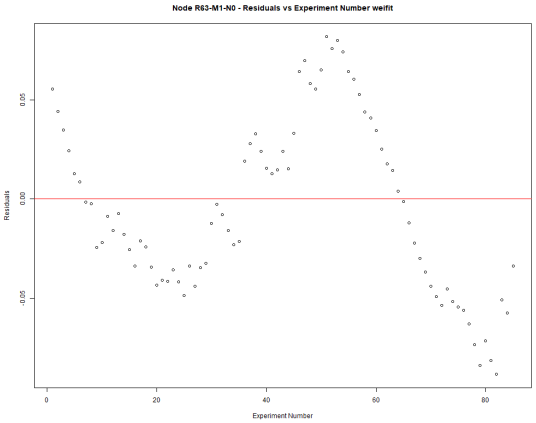
(a) Residuals vs Experiment Number - Node R62-M0-N0 expfit



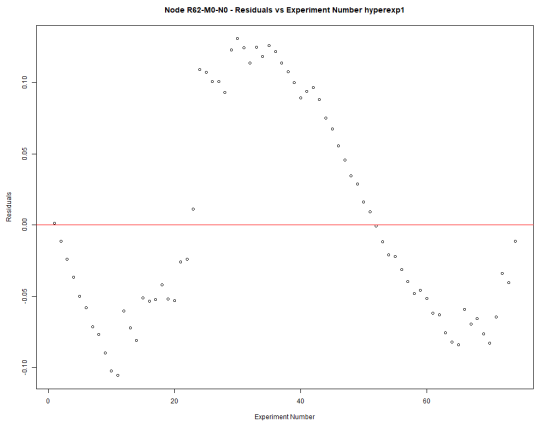
(b) Residuals vs Experiment Number - Node R63-M1-N0 expfit



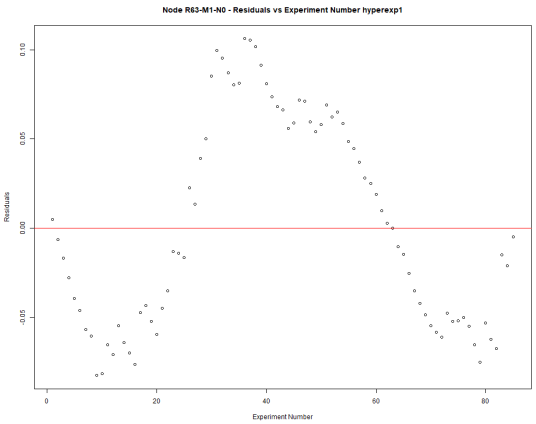
(a) Residuals vs Experiment Number - Node R62-M0-N0 weifit



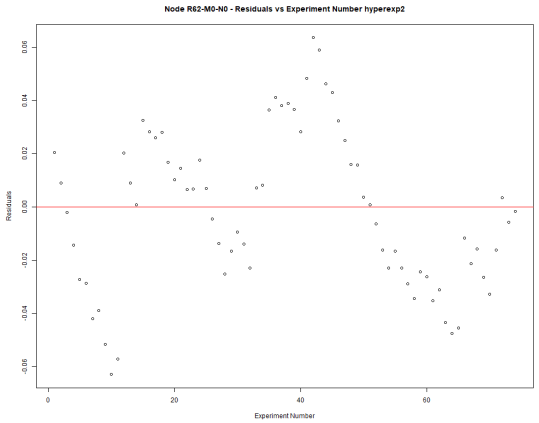
(b) Residuals vs Experiment Number - Node R63-M1-N0 weifit



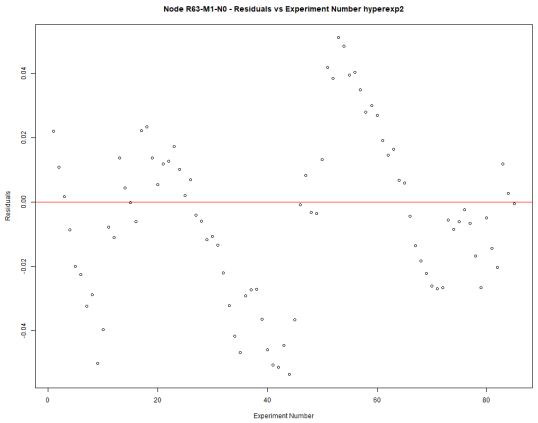
(a) Residuals vs Experiment Number - Node R62-M0-N0 hyperexp1



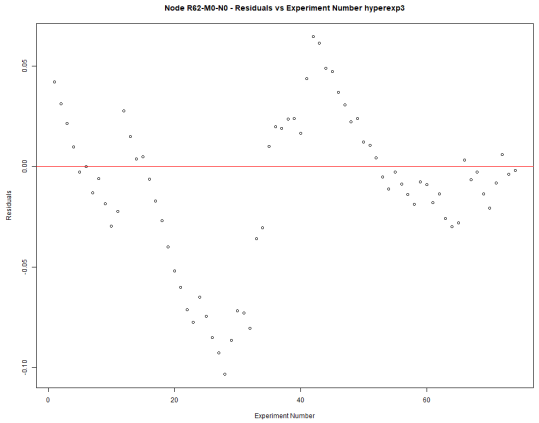
(b) Residuals vs Experiment Number - Node R63-M1-N0 hyperexp1



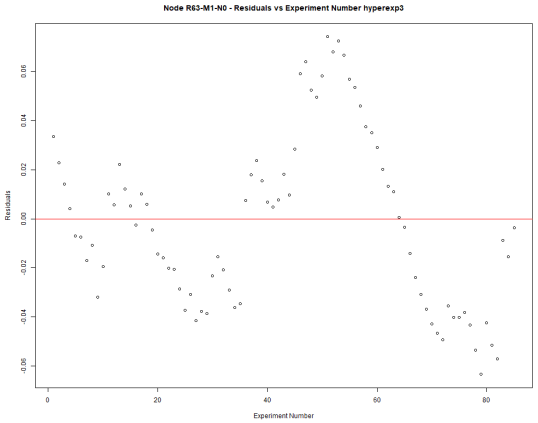
(a) Residuals vs Experiment Number - Node R62-M0-N0 hyperexp2



(b) Residuals vs Experiment Number - Node R63-M1-N0 hyperexp2



(a) Residuals vs Experiment Number - Node R62-M0-N0 hyperexp3



(b) Residuals vs Experiment Number - Node R63-M1-N0 hyperexp3

Anche dai grafici dei residui sulle osservazioni si può evincere che esistono dei trend nei residui. Questo è indice del fatto che esistono fenomeni interni non modellati correttamente e che si dovrebbe variare il modello per tenerne conto. Questa condizione è valida per tutti i nodi analizzati e per tutti i modelli di regressione utilizzati.

4 Conclusioni

Al termine dell'analisi è possibile fare una summa di quanto emerge dai dati e del lavoro fatto. Di seguito, le tabelle contenenti le sintesi dei risultati delle analisi effettuate sui nodi e sui file di settembre e ottobre:

R62-M0-N0

Coalescence Window	100 s	Tuple count	75
MTTF	Mean: 69091.26 SD: 97382.51 Median: 41945 SIQR: 46143.5	MTTF (90% confidence) MTTF (95% confidence)	[50231.37,87951.14] [46529.56,91652.95]
Exponential model parameters	λ:0.0000168003	KS Test	p-value: 0.008718 reject: yes
Weibull model parameters	λ: 0.00001975111 α: 0.45526628071	KS Test	p-value: 0.7838 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1: 0.000293166159 α_2: 0.5 λ_2: 0.000006384861	KS Test	p-value: 0.5117 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1: 0.00117699957 α_2: 0.7 λ_2: 0.00001028589	KS Test	p-value: 0.9964 reject: no
Hyperexponential model 3 parameters	α_1: 0.2 λ_1: 0.003187496050 α_2: 03 λ_2: 0.000008563631 α_3: 0.5 λ_3: 0.000015386966	KS Test	p-value: 0.7838 reject: no

(a) Node R62-M0-N0

R62-M0-N4

Coalescence Window	100 s	Tuple count	77
MTTF	Mean: 67274.16 SD: 96735.14 Median: 36404 SIQR: 46824.75	MTTF (90% confidence) MTTF (95% confidence)	[48794.13,85754.19] [45169.23,89379.09]
Exponential model parameters	λ:0.0000180472	KS Test	p-value: 0.01008 reject: no
Weibull model parameters	λ: 0.00002104002 α: 0.44879589648	KS Test	p-value: 0.7973 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1:0.00031671755 α_2: 0.5 λ_2:0.00000665789	KS Test	p-value: 0.5291 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1: 0.00134273269 α_2: 0.7 λ_2: 0.00001074345	KS Test	p-value: 0.997 reject: no
Hyperexponential model 3 parameters	α_1: 0.2 λ_1: 0.003564331440 α_2: 03 λ_2: 0.000006872258 α_3: 0.5 λ_3: 0.000019237264	KS Test	p-value: 0.7973 reject: no

(b) Node R63-M1-N0

R62-M0-NC

Coalescence Window	100 s	Tuple count	76
MTTF	Mean: 68170.39 SD: 96975.28 Median: 40402 SIQR: 45466.75	MTTF (90% confidence) MTTF (95% confidence)	[49518.23,86822.54] [45858.40,90482.37]
Exponential model parameters	λ:0.00001720933	KS Test	p-value: 0.009386 reject: yes
Weibull model parameters	λ: 0.00002032599 α: 0.45337274428	KS Test	p-value: 0.7907 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1:0.000278342672 α_2: 0.5 λ_2:0.000006532374	KS Test	p-value: 0.5204 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1: 0.00113002532 α_2: 0.7 λ_2: 0.00001048287	KS Test	p-value: 0.9967 reject: no
Hyperexponential model 3 parameters	α_1: 0.2 λ_1: 0.00311851975 α_2: 03 λ_2: 0.00000795561 α_3: 0.5 λ_3: 0.00001663795	KS Test	p-value: 0.6562 reject: no

(c) Node R63-M1-N0

R63-M1-N0

Coalescence Window	100 s	Tuple count	87
MTTF	Mean: 59484.21 SD: 90851.26 Median: 20435 SIQR: 42358.5	MTTF (90% confidence) MTTF (95% confidence)	[40005.66,78962.76] [43192.44,75775.98]
Exponential model parameters	λ:0.00002247204	KS Test	p-value: 0.006527 reject: yes
Weibull model parameters	λ: 0.00002546706 α: 0.44985073126	KS Test	p-value: 0.8488 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1:0.000314395547 α_2: 0.5 λ_2:0.000007823346	KS Test	p-value: 0.6016 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1: 0.00111021031 α_2: 0.7 λ_2: 0.00001252088	KS Test	p-value: 0.9987 reject: no
Hyperexponential model 3 parameters	α_1: 0.2 λ_1: 0.002290443042 α_2: 03 λ_2: 0.0000794123345 α_3: 0.5 λ_3: 0.000008343108	KS Test	p-value: 0.9376 reject: no

(d) Node R63-M1-N0

R63-M1-N8

Coalescence Window	100 s	Tuple count	87
MTTF	Mean: 59486.56 SD: 92361.04 Median: 19959 SIQR: 44346	MTTF (90% confidence) MTTF (95% confidence)	[42924.05,76049.07] [39684.31,79288.81]
Exponential model parameters	λ: 0.00002344962	KS Test	p-value: 0.002166 reject: yes
Weibull model parameters	λ: 0.00002684208 α: 0.42640502463	KS Test	p-value: 0.8488 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1:0.00046207772 α_2: 0.5 λ_2:0.000007888606	KS Test	p-value: 0.4773 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1: 0.00154214998 α_2: 0.7 λ_2: 0.00001285185	KS Test	p-value: 0.985 reject: no
Hyperexponential model 3 parameters	α_1: 0.2 λ_1: 0.003468257305 α_2: 03 λ_2: 0.000005425371 α_3: 0.5 λ_3: 0.000033187577	KS Test	p-value: 0.8488 reject: no

(e) Node R63-M1-N8

R63-M1-NC

Coalescence Window	100 s	Tuple count	89
MTTF	Mean: 58134.15 SD: 96315.19 Median: 18295.5 SIQR: 40149	MTTF (90% confidence) MTTF (95% confidence)	[41064.28,75204.01] [37726.91,78541.39]
Exponential model parameters	λ: 0.00002393073	KS Test	p-value: 0.002542 reject: yes
Weibull model parameters	λ: 0.00002785695 α: 0.43314939364	KS Test	p-value: 0.8585 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1:0.000399910293 α_2: 0.5 λ_2:0.000008358531	KS Test	p-value: 0.6164 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1: 0.00138153276 α_2: 0.7 λ_2: 0.00001342291	KS Test	p-value: 0.9868 reject: no
Hyperexponential model 3 parameters	α_1: 0.2 λ_1: 0.003039837495 α_2: 03 λ_2: 0.000095913323 α_3: 0.5 λ_3: 0.000008841894	KS Test	p-value: 0.943 reject: no

(f) Node R63-M1-NC

bglsep_1

Coalescence Window	120 s	Tuple count	201
MTTF	Mean: 12667.81 SD: 20501.46 Median: 4676.5 SIQR: 6806.5	MTTF (95% confidence) MTTF(90% confidence)	[9809.119,15526.501] [10272.16,15063.46]
Exponential model parameters	λ:0.0001312641	KS Test	p-value = 0.005657 reject: yes
Weibull model parameters	λ: 0.0001160867 α: 0.6092869988	KS Test	p-value: 0.7024 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1: 0.00051778815 α_2: 0.5 λ_2: 0.00004228989	KS Test	p-value: 0.9186 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1:0.00106311850 α_2: 0.7 λ_2: 0.00006820458	KS Test	p-value = 0.9615 reject: no
Hyperexponential model 3 parameters	α_1: 0.5 λ_1: 0.00004827449 α_2: 0.3 λ_2: 0.00096294941 α_3: 0.2 λ_3: 0.00017086151	KS Test	p-value = 0.9969 reject: no

(a) bglsep_1

bgloct_1

Coalescence Window	100 s	Tuple count	135
MTTF	Mean: 19696.52 SD: 25623.61 Median: 8835 SIQR: 12261.5	MTTF (95% confidence) MTTF(90% confidence)	[15318.22,24074.82] [16030.03,23363.01]
Exponential model parameters	λ: 0.00007235459	KS Test	p-value: 0.1325 reject: no
Weibull model parameters	λ: 0.00006507445 α: 0.62966960380	KS Test	p-value: 1 reject: no
Hyperexponential model 1 parameters	α_1: 0.5 λ_1: 0.00024554340 α_2: 0.5 λ_2: 0.00002526216	KS Test	p-value: 0.921 reject: no
Hyperexponential model 2 parameters	α_1: 0.3 λ_1:0.00056017654 α_2: 0.7 λ_2: 0.00003839712	KS Test	p-value = 0.9928 reject: no
Hyperexponential model 3 parameters	α_1: 0.5 λ_1: 0.00003015863 α_2: 0.3 λ_2: 0.00052811616 α_3: 0.2 λ_3: 0.00007194833	KS Test	p-value = 0.9993 reject: no

(b) bgloct_1

E' interessante notare come il modello con il p-value più alto sia l'iperesponenziale a due termini per i nodi, mentre per i mesi interi sia l'iperesponenziale a tre termini. Volendo fare una ipotesi puramente speculativa, potrebbe essere che l'analisi del mese intero richieda di rappresentare nel modello l'andamento di più fenomeni sovrapposti, corrispondenti ai vari comportamenti che i nodi diversi hanno rispetto all'insorgenza delle failures. In questo caso quindi un modello più complesso, a tre termini, rappresenta meglio i fenomeni osservati.
In generale i modelli molto semplici come l'esponenziale non sembrano adatti a modellare il fenomeno in nessuna delle istanze analizzate, poiché pur passando il test di KS hanno un p-value molto inferiore agli altri modelli. Questa considerazione viene confermata ulteriormente dalle analisi tramite grafici quantile-quantile, che rivelano come i modelli expfit, weifit e hyperexp1 siano molto distanti da un andamento considerato ottimale.
Una ulteriore considerazione riguarda i grafici dei residui rispetto alla risposta predetta e alle osservazioni: per tutti i modelli e per tutti i nodi e mesi si è osservato come siano presenti dei trend nei residui. La mancanza di un andamento randomico fa intuire che potrebbero esserci dei fenomeni che non sono modellati accuratamente dai modelli proposti e che si potrebbe beneficiare positivamente da un nuovo tipo di modello che possa migliorare questo aspetto.
In ultimo si può osservare come nel mese di settembre ci siano degli itemset frequenti, ottenuti analizzando le transazioni estratte dai log, e che la situazione che si verifica più spesso è che nello stesso rack falliscano più nodi insieme, sintomo di una forte dipendenza tra alcuni nodi dello stesso rack.