Queste brevi note prendono spunto dall'articolo di K.G. Russell, dal titolo "*Estimating the Value of e by Simulation*", pubblicato nel 1991 sulla rivista *The American Statistician* (volume n. 45, pagine 66–68). Questo articolo illustra non solo, come preannunciato nel titolo, un modo per stimare il valore del numero di Nepero *e* (2.718…) impiegando la simulazione, ma anche come andare un po' oltre l'approccio, per così dire, *naive* al metodo Monte Carlo, volutamente impiegato nel nostro testo didattico. Per ulteriori approfondimenti il lettore è invitato a leggere il testo *on line* su queste stesse pagine *web* (capitolo 6) che introduce alla problematica della valutazione della dimensione di una simulazione.

L'articolo di Russell comincia proponendo il seguente "semplice" problema: **in media**, quanti numeri pseudo-casuali uniformemente distribuiti fra zero e uno è necessario generare affinché la loro somma **ecceda** il valore uno? Una risposta ragionevole sembrerebbe essere "due", ma in questo caso la "ragionevolezza" non si rivela buona consigliera. Infatti, se è vero che come minimo è necessario generare due numeri, è altresì vero che, di tanto in tanto, sarà necessario un numero più elevato di estrazioni, di modo che la media sarà superiore a 2. Lo *script* che segue (che proponiamo senza commento, in quanto non dissimile da tanti altri che trovate nel testo) produce una approssimazione Monte Carlo della corrispondente distribuzione di probabilità.

```
> nrep <- 100000
> out <- rep(0,nrep)
> for (i in 1:nrep)
+ {
+    s <- 0
+    n <- 0
+    while (s<=1) {
+       n <- n+1
+       x <- runif(1)
+       s <- s+x
+    }
+    out[i] <- n
+ }
> table(out)
out
    2     3     4     5     6     7     8     9
50147 33390 12409  3238   693   103    18     2
```

Come si vede, più di metà delle volte bastano proprio due numeri, ma circa una volta su 3 sono necessarie tre estrazioni e, qualche volta (2 su un totale di 100000), ne sono state necessarie ben nove.

Ma la cosa forse più interessante è che la *media* di questa distribuzione (quella vera, beninteso) è proprio il "mitico" numero *e* (per la dimostrazione analitica si rimanda all'articolo di Russell). Naturalmente noi possiamo richiedere una verifica empirica, calcolando la media delle nostre 100000 repliche:

```
> xm <- mean(out)
> xm
[1] 2.71331
```

Impiegando il valore ottenuto come una *stima* del valore vero (2.718…), possiamo calcolare un intervallo di confidenza al 95%, ricorrendo alla formula riportata a pag. 236 del testo, stimando la varianza a partire dai risultati della simulazione.

```
> se <- sqrt(var(out)/nrep)
> cil <- xm - 1.96*se
> cih <- xm + 1.96*se
> cat(xm,cil,cih,(cih-cil),"\n")
2.71331 2.707917 2.718703 0.01078510
```

La prima riga stima l'*errore standard* a partire, appunto, dalla varianza campionaria (ottenuta richiamando la funzione `var`) e le due successive calcolano l'estremo inferiore (`cil`) e quello superiore (`cih`) dell'intervallo di confidenza al 95%. Naturalmente il lettore attento si sarà accorto che stiamo impiegando l'approssimazione normale, dal momento che la media in parola si basa su un campione di dimensione 100000 (che possiamo ritenere un valore sufficientemente elevato). L'ultima riga, oltre a mostrare la media prodotta dalla simulazione, fa vedere che la media vera dovrebbe essere compresa, con una confidenza del 95%, fra 2.708 e 2.719 (e, in effetti, il numero *e* è proprio compreso in questo intervallo). L'ultima informazione (`0.01078510`) riguarda l'ampiezza dell'intervallo di confidenza e qui cominciano le note dolenti, nel senso che la nostra stima non appare molto precisa. Se volessimo portare l'ampiezza a 0.001 (dieci volte inferiore), dovremmo (vedi quanto scritto nella pagina *web* relativa al capitolo 6) aumentare di 100 volte il numero delle repliche (quindi da 100000 a dieci milioni).

Tuttavia esistono opportuni accorgimenti (che potremmo considerare delle varianti al metodo Monte Carlo, per così dire, *naive*) che permettono di ottenere una maggiore precisione (quindi ridurre l'ampiezza dell'intervallo di confidenza) senza dovere aumentare in maniera così "spropositata" il numero delle repliche. Ad esempio, è possibile impiegare il metodo delle *variabili antitetiche*, che viene illustrato proprio nell'articolo di Russell. Questo metodo è implementato nello *script* che segue, che vale la pena di descrivere con qualche dettaglio (sempre rimandando per gli opportuni approfondimenti metodologici all'articolo in parola).

```
> nrep <- 100000
> ris <- matrix(0,nrow=nrep,ncol=2)
> for (i in 1:nrep)
+ {
+    s1 <- 0; n1 <- 0
+    s2 <- 0; n2 <- 0
+    while (s1<=1 | s2<=1) {
+       u1 <- runif(1)
+       u2 <- 1 - u1
+       if (s1<=1) {
+          s1 <- s1 + u1
+          n1 <- n1 + 1
+       }
+       if (s2<=1) {
+          s2 <- s2 + u2
+          n2 <- n2 + 1
+       }
+    }
+    ris[i,1] <- n1
+    ris[i,2] <- n2
+ }
> out <- apply(ris,1,mean)
```

Come si vede, oltre ad estrarre un numero `u1`, uniformemente distribuito fra 0 e 1, se ne considera anche un secondo, `u2`, calcolato come differenza fra 1 e `u1`. La cosa può apparire strana, dal momento che il secondo è assolutamente determinato, una volta che si conosca il primo, ma l'aspetto importante è che i due numeri sono **negativamente** correlati. I due valori estratti sono sommati, separatamente l'uno dall'altro, ai valori precedentemente estratti, proseguendo l'estrazione finché i totali associati non diventano entrambi maggiori di uno. Nella matrice `ris` si memorizza quante estrazioni sono state necessarie perché la somma associata ad `u1` diventasse maggiore di uno (prima colonna) e quante estrazioni sono state necessarie perché la somma associata ad `u2` diventasse maggiore di uno (seconda colonna). A questo punto facciamo, riga per riga (quindi, replica per replica) la media aritmetica dei due valori, che memorizziamo nel vettore `out`; questi valori ci permetteranno di stimare *e* con una maggiore precisione. Infatti, ripetendo le ultime righe dello *script* già presentato in precedenza, troviamo:

```
> xm <- mean(out)
> se <- sqrt(var(out)/nrep)
> cil <- xm - 1.96*se
> cih <- xm + 1.96*se
> cat(xm,cil,cih,(cih-cil),"\n")
2.718965 2.716776 2.721154 0.004378011
```

La media vera (il numero *e*) dovrebbe essere compreso, con una confidenza del 95%, fra 2.717 e 2.721 (ancora una volta il numero *e* è compreso in questo intervallo), ma l'ampiezza dell'intervallo di confidenza è passato, con lo stesso numero di repliche, a 0.004.
Il lettore attento obietterà a questo punto, e a ragione, che, però, il numero di valori estratti è stato certamente superiore a quello precedente (ottenuto con il primo *script*, quello *naive*) e che era stato pari a 271331. Vediamo quante estrazioni abbiamo dovuto fare:

```
> sum(apply(ris,1,max))
[1] 343793
```

Effettivamente abbiamo fatto 343793 estrazioni, un valore superiore di 1.267 volte a quello precedente (circa un quarto in più), ma al contempo siamo riusciti a passare da una ampiezza pari a circa 0.01 ad una ampiezza pari a circa 0.0044. In altre parole, l'ampiezza si è ridotta di oltre due volte senza aver dovuto quadruplicare la dimensione del campione. Si è potuto raggiungere tale risultato proprio perché abbiamo impiegato **variabili negativamente correlate** (*antitetiche*); infatti, in questo caso, la varianza di una somma di due variabili casuali è la somma delle rispettive varianze **diminuita** del doppio del valore della covarianza (che nel caso in questione è massima, dal momento che la loro correlazione è –1).