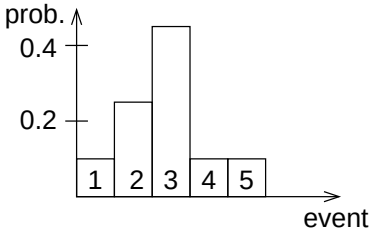
Prof. Dr. A.K. Hartmann
WiSe 2022/23, 5.04.4521
Blatt 6, 24.2.2023

Übungen zur Computerorientierten Physik

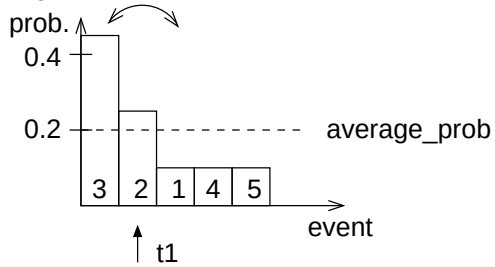
6 Ziehen diskreter Zufallszahlen

Betrachten Sie den Fall von N diskreten Ereignissen (o.B.d.A. hier $1, 2, \ldots, N$) mit Wahrscheinlichkeiten $p_i \geq 0$ und $\sum_i p_i = 1$. Wie kann man effizient Zufallszahlen gemäß den $\{p_i\}$ ziehen? Hier ein Beispiel für $N = 5$ mit $p_1 = 0.1$, $p_2 = 0.25$, $p_3 = 0.45$, $p_4 = 0.1$, $p_5 = 0.1$:

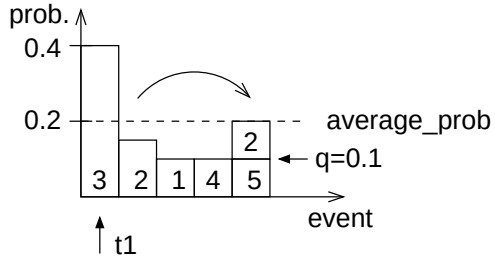


Die trivialste Variante ist, die Zurückweisungsmethode zu verwenden. Falls sich die p_i aber stark unterscheiden, kann das beliebig langsam sein.
Die nächst-einfachste Variante ist, das Intervall $[0, 1)$ in Teilintervalle aufzuteilen, so dass jedem Ereignis i ein Teilintervall I_i der Breite p_i zugeordnet wird, also hier z.B., $I_1 = [0, 0.1)$, $I_2 = [0.1, 0.35)$, $I_3 = [0.35, 0.8)$, $I_4 = [0.8, 0.9)$, $I_5 = [0.9, 1)$. Dann zieht man eine Zufallszahl r gemäß $U(0, 1)$ und wählt das Ergnis i mit $r \in I_i$. Dazu muss in der einfachsten Variante aber jedesmal ein Array der Teilintervalle durchgegangen werden (es reicht die oberen Grenzen zu speichern). Dies dauert $O(N)$.
Schneller wird es, wenn man die Teilintervalle iterativ in zwei Teile teilt, und immer in der Halbmenge weitersucht, die die Zufallszahl r enthält. Das dauert $O(\log N)$.
Noch schneller, in $O(1)$ geht es mit folgendem Verfahren, dass auf Walker zurück geht. Die Grundidee ist, die Ergeinisse so auf gleiche "Päckchen" zu verteilen, dass jedes Päckchen genau einen Anteil $p_{\rm avg} = 1/N$ (genannt `average_prob` im Programm) entspricht, und in jedem Päcken maximal zwei Ereignisse beheimatet sind. (So sind manche Ereignisse in mehreren Päcken, insbesondere falls $p_i > p_{\rm avg}$.) Man muss sich dann für jedes Päckchen i nur merken, welche beiden Eregnisse a_i und b_i repräsentiert werden, und welchen Anteil q_i dem Ereignis a_i zugeordnet ist (Ereignis b_i hat dann Anteil $1 - q_i$). Diese Werte werden in einer gemeinsamen Datenstruktur ("Tabelle") gespeichert (s.u.). Beim Ziehen eines Ereignisses zieht man dann einfach eines der Päckchen i mit Gleichverteilung (eine Zufallszahl nötig) und danach bestimmt man eine zweite Zufallszahl r. Falls $r < q_i$ wird das Ereignis a_i gewählt, sonst b_i.
Algorithmisch geht man beim Aufbau der Tabelle wie folgt vor: Im Päcken q_i wird das Ereignis i gespeichert ($a_i = i$) und $q_i = p_i$ initalisiert. Während der Berechnung der Tabelle gibt q_i an, wieviel Wahrscheinlichkeit für das Ergeignis a_i noch in dem Päckchen i repräsentiert wird. Zunächst werden die Ereignisse so umsortiert, dass erst alle Ereignisse mit $p_i > p_{\rm avg}$ aufgeführt sind, dann der Rest ($p_i \leq p_{\rm avg}$) ("Eigenschaft S") Das erfolgt, indem man mit zwei Indexzählern `t0` und `t1` links (1. bzw. 0 bei C Array) und rechts (N bzw. $N - 1$ im C Array)

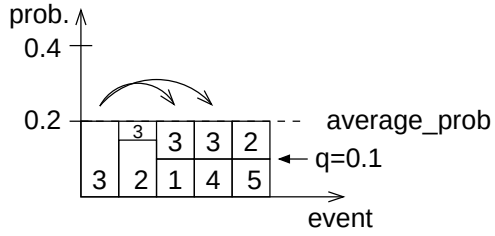
anfängt und iterativ beide Zeiger unabhängig voneinander nach rechts bzw. links verschiebt, bis man jeweils auf ein nicht gemäß "S" einsortiertes Element trifft. Diese beiden Elemente werden dann getauscht. Das macht man solange, bis `t0` und `t1` aneinander vorbei gelaufen sind (also bis `t0` > `t1`). Danach zeigt `t1` auf das am weitesten rechts stehende Ereignis, das eine Wahrscheinlichkeit $> p_{\text{avg}}$ aufweist. Das Ergebnis für das Beispiel sieht wie folgt aus:



Nun werden, beginnend ganz rechts und nach links laufend ($t = N, N-1, \ldots, 2$, bzw. für C Arrays $N-1, \ldots, 1$), die Päcken immer soweit aufgefüllt, dass sie insgesamt die Wahrscheinlichkeit p_{avg} repräsentieren. Dazu wird ein entsprechender Anteil immer dem Päcken `t1` entnommen, also $q_{\texttt{t1}}$ wird um $(p_{\text{avg}} - q_t)$ reduziert. Da $q_{\texttt{t1}} > p_{\text{avg}}$ wird nach der Reduzierung der Anteil des Päckens `t1` in jedem Fall positiv sein, aber ggf. kleiner als p_{avg}. Im letzteren Fall wird dann das Päckchen automatisch zu der Menge der Päckchen, die wieder aufgefüllt werden müssen. Dazu wird der Index `t1` einen nach links verschoben (also `t1` um eins reduziert). Für das Beispiel sieht das nach dem ersten Schritt wir folgt aus:



Hinweis: Das am weitesten links stehende Päcken hat zum Schluss automatisch den Anteil p_{avg} und `t1` steht dann links davon (also im undefinierten Bereich, was aber dann egal ist).
Die Endsituation für das Beispiel sieht wie folgt aus (hier ist $q_1 = 0.2$, $q_2 = 0.15$ und $q_3 = q_4 = q_5 = 0.1$. Im Allgemeinen können die Werte für q_i aber alle verschieden sein, es gilt aber immer $q_i \leq p_{\text{avg}}$):



Ganz zum Schluss werden die Wahrscheinlichkeiten q_i von globalen Wahrscheinlichkeiten zu relative Wahrscheinlichkeiten umnormiert, so dass Sie nur noch den Anteil des Ereignisses a_i im Päckchen i repräsentieren. Dazu müssen nur alle Werte q_i durch p_{avg} geteilt werden (siehe bedingte Wahrscheinlichkeiten $p(a|b) = p(a,b)/p(b)$).
Ihre Aufgabe ist, die Walker Methode teilzuimplementieren und zu testen:

Laden Sie dazu `discrete_variate_fragment.c` vom StudIP. Das Programm enthält:

- Eine Datenstruktur `discrete_variate_t` zur Speicherung der Tabelle.
- Ein Rahmen für eine Funktion `setup_table()`, die die Tabelle in $O(N)$ initialisiert.
- Ein Rahmen für eine Funktion `draw_number()`, die ein Ereignis gemäß der vorgegebenen Verteilung zieht.
- Ein Hauptprogramm, dass eine abgeschnittene Poisson Verteilung mit Parameter `mu` (1. Aufrufparameter) vorgibt und `num_runs` (2. Aufrufparameter) mal eine Zufallszahl zieht und daraus ein Histogramm bildet.

  Hinweis: Bei `num_run` $= 0$ wird die original Verteilung ausgegeben.

Aufgaben:

- Verstehen Sie das vorhandene Programm.
- Vervollständigen Sie die Funktionen:

  `setupt_table()`, (5 Punkte)

  und `draw_number()`, (2 Punkte)

  so dass man korrekt gemäß der vorgegebenen Verteilung zieht.
- Führen Sie Simulationen für `mu=5` und `num_runs` $= 0, 10^2, 10^4$ und 10^6 durch und leiten Sie die Ausgabe in Dateien um. Vergleichen Sie die Histogramme per `gnuplot` mit der Originalverteilung. (2 Punkte)

  Verwende Sie auch andere Parameter für `mu`. Was müssen Sie für große Werte von `mu` im Hauptprogramm ändern? (1 Punkt)