The RWiener Package: an R Package Providing Distribution Functions for the Wiener Diffusion Model

by Dominik Wabersich and Joachim Vandekerckhove

Abstract We present the **RWiener** package that provides R functions for the Wiener diffusion model. The core of the package are the four distribution functions `dwiener`, `pwiener`, `qwiener` and `rwiener`, which use up-to-date methods, implemented in C, and provide fast and accurate computation of the density, distribution, and quantile function, as well as a random number generator for the Wiener diffusion model. We used the typical Wiener diffusion model with four parameters: boundary separation, non-decision time, initial bias and drift rate parameter. Beyond the distribution functions, we provide extended likelihood-based functions that can be used for parameter estimation and model selection. The package can be obtained via CRAN.

Introduction

The diffusion model is one of the most successful models of choice reaction time in cognitive psychology (see Wagenmakers, 2009, for an overview) and, while software packages that make diffusion model analyses possible have been made available (Vandekerckhove and Tuerlinckx, 2007, 2008; Vandekerckhove et al., 2011; Voss and Voss, 2007; Wagenmakers et al., 2007; Wiecki et al., 2013), a diffusion model R package was so far missing. Due to the nature of the model, no closed analytical forms for the computation of the distribution functions exist, which has generated a need for more involved computational approximations of the various functions (Blurton et al., 2012; Navarro and Fuss, 2009; Tuerlinckx et al., 2001).

Here, we present an R package, **RWiener**, that provides the four functions R uses to represent a distribution, implemented for the Wiener diffusion model: the *density function* `d` (Navarro and Fuss, 2009) to compute the probability density function (PDF) at a given quantile for a given parameter set; the *probability function* `p` (Blurton et al., 2012) to compute the cumulative distribution (CDF) at a given quantile for a given parameter set; the inverse CDF or *quantile function*[1] `q` to compute the quantile at a given p-value (CDF-value) for a given parameter set; and a *random number generator* (RNG) `r` (Tuerlinckx et al., 2001) to generate random samples for a given parameter set.

In addition, we extended the package to include several likelihood-based functions that can be used in combination with R's built-in estimation routines, like the `nlm` or `optim` function, to estimate the model parameters for an observed data set, assuming a Wiener diffusion process as the underlying process that created the data.

First, we will present the standard Wiener diffusion model with four parameters: the boundary separation α, the non-decision time τ, the bias parameter β and a drift rate parameter δ. Next, we will show the standard functionality of the **RWiener** package and how it can be used for parameter estimation. To conclude, we add a small discussion about the utility and extensibility of the here presented methods.

The Wiener diffusion model

For two-choice reaction time (2CRT) data coming from experiments in which subjects are asked to give one of two responses on a decision task, the Wiener diffusion model and its extensions have provided useful modeling approaches. In its simplest complete form, the Wiener diffusion model includes four parameters. The decision process is thought of as a continuous random walk, or diffusion, process that starts somewhere between two boundaries and ends as soon as it hits or crosses one or the other. The time of the first passage, and which boundary is hit first (and therefore which decision will be made) is probabilistic with a probability determined by the parameter set.

The standard Wiener diffusion model incorporates the following four parameters, a summary of which is provided in Table 1: (1) the boundary separation α which defines the distance between the two boundaries, with the first boundary being at 0 and the second being at α, (2) the non-decision time τ that defines the time that passes without any diffusion process going on (e.g., this incorporates the time needed to encode a stimulus and to execute a motor response), (3) the bias parameter β defining

[1] Implemented using a reverse lookup algorithm of the CDF function.

Symbol	Parameter	Interpretation
α	Boundary separation	Speed-accuracy trade-off (high α means high accuracy)
β	Initial bias	Bias for either response ($\beta > 0.5$ means bias towards response 'A')
δ	Drift rate	Quality of the stimulus (close to 0 means ambiguous stimulus)
τ	Nondecision time	Motor response time, encoding time (high means slow encoding, execution)

Table 1: The four main parameters of the Wiener diffusion model, with their substantive interpretations. Reprinted with permission from Vandekerckhove (2009). Copyright 2009, Joachim Vandekerckhove and Department of Psychology and Educational Sciences, University of Leuven, Belgium.

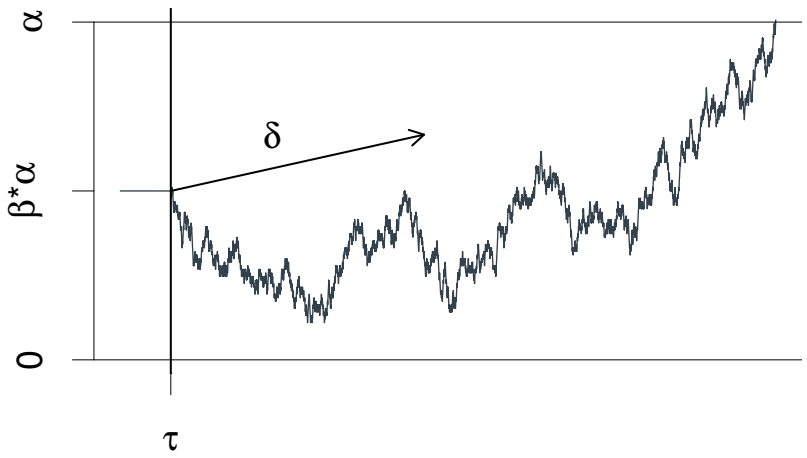

Figure 1: A graphical illustration of the Wiener diffusion model for two-choice reaction times. An evidence counter starts at value $\alpha\beta$ and evolves with random increments. The mean increment is δ. The process terminates as soon as the accrued evidence exceeds α or deceeds 0. The decision process starts at time τ from the stimulus presentation and terminates at the reaction time.

the relative starting point of the diffusion process between the two boundaries (the absolute starting point ζ_0 can be obtained by $\zeta_0 = \beta\alpha$) and (4) the drift rate parameter δ which captures the tendency of the diffusion process to drift towards the upper boundary. Given these parameters, the latent information accumulation process is modeled with the stochastic process $\frac{d}{dt}X_t \sim \text{Normal}\left(\delta, s^2\right)$, with initial condition $X_0 = \alpha\beta$. The decision time is modeled as the first time at which $X_t \leq 0$ or $X_t \geq \alpha$, and the response time is the sum of the decision time and the non-decision time τ. Figure 1 shows an illustration of the Wiener diffusion model as described.

The parameters have the following restrictions: $0 < \beta < 1$, $\alpha > 0$, $\tau > 0$. Often, the variance parameter s is fixed at 0.1, whereas we fixed it at 1, yielding a more convenient interpretation of the drift rate parameter and greater computational efficiency. Conversion to a scale with the variance parameter fixed at any s can be done by multiplying the α and δ parameters by s.

The upper boundary may be defined as corresponding to correct trials, with the lower boundary corresponding to error trials. However, it often also makes sense to map qualitatively different responses to the two boundaries (e.g., in a same-different discrimination task, the upper bound could represent the 'different' responses). For generality, we will always speak simply of upper and lower boundaries. An overview of the computational strategy for the PDF and CDF functions is provided in the Appendix.

Usage of the package

The package can be obtained via CRAN for R version 2.15.0 or higher, and includes documentation that provides useful information and examples. As a guide through this demonstration of the **RWiener** package functionality, we use an example data set created by the random function of the package (using a RNG seed of 0 for exact reproducibility):

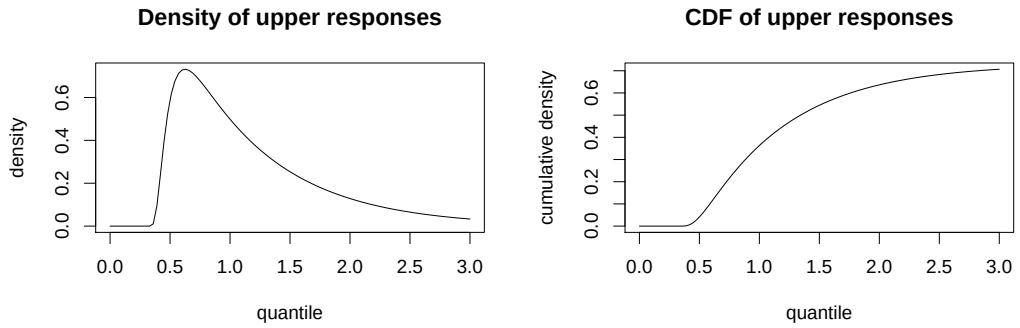

Figure 2: Example of plots made with function `dwiener` and `pwiener`.

```
set.seed(0)
dat <- rwiener(n=100, alpha=2, tau=.3, beta=.5, delta=.5)
```

The arguments for this function call are explained in the next paragraph. This command will generate a random data set, consisting of 100 observations, each generated from a random Wiener diffusion model process with the given four parameters. We used a drift rate parameter δ of 0.5, meaning that the drift diffusion process slightly tends towards the upper boundary. There was no initial bias towards either of the two boundaries, indicated by $\beta = 0.5$. The boundaries are separated by a value of $\alpha = 2$ and we used a non-decision time of $\tau = 0.3$, so there will never be a RT smaller than 0.3.

The created `dat` object is a dataframe with 100 observations and two variables: `dat$q` contains the RTs, whereas `dat$resp` contains a factor to indicate the response made (*upper* vs. *lower*). Note that the upper bound is assigned to the first level and the lower bound to the second one.

The four main functions of the package are described in the following subsection. All four functions are implemented in C, using the R library for C functions. These C functions are called inside end-user R functions, so that the end-user never calls the C functions directly, only indirectly through the provided R functions.

Standard distribution functions

To get the density of a specific quantile, one can use the density function `dwiener`:

```
dwiener(dat$q[1], alpha=2, tau=.3, beta=.5, delta=.5, resp=dat$resp[1], give_log=FALSE)
```

The function takes seven arguments, the first argument is the RT at which the density shall be evaluated, the next four arguments are the model parameters α, τ, β, and δ. The next argument, `resp` takes any of three valid values: *upper*, *lower*, or *both*, meaning that the function shall return the density at the given quantile for the upper boundary, the lower boundary or the sum of both densities together, respectively. Its default value is set to *upper*. Lastly, the function can be given an additional argument `give_log` that takes *TRUE* or *FALSE* as values and is set to *FALSE* as a default. When this argument is *TRUE*, the function will return the logarithm of the density instead of the density itself.

This function can also be used to draw simple plots. For example, the following code results in the plot in the left panel of Figure 2:

```
curve(dwiener(x, 2, .3, .5, .5, rep("upper", length(x))),
      xlim=c(0,3), main="Density of upper responses",
      ylab="density", xlab="quantile")
```

To calculate the cumulative distribution, we can use the CDF function `pwiener` as follows:

```
pwiener(dat$q[1], alpha=2, tau=.3, beta=.5, delta=.5, resp=dat$resp[1])
```

The arguments are the same as for the previously described function, without the `give_log` argument. Further, this function can be used in the same manner to draw plots, like the one shown in the right panel of Figure 2.

We can also find the appropriate quantile for a given probability via the inverse CDF function, implemented as `qwiener`:

```
# lookup of the .2 quantile for the CDF of the lower boundary
qwiener(p=.2, alpha=2, tau=.3, beta=.5, delta=.5, resp="lower")
```

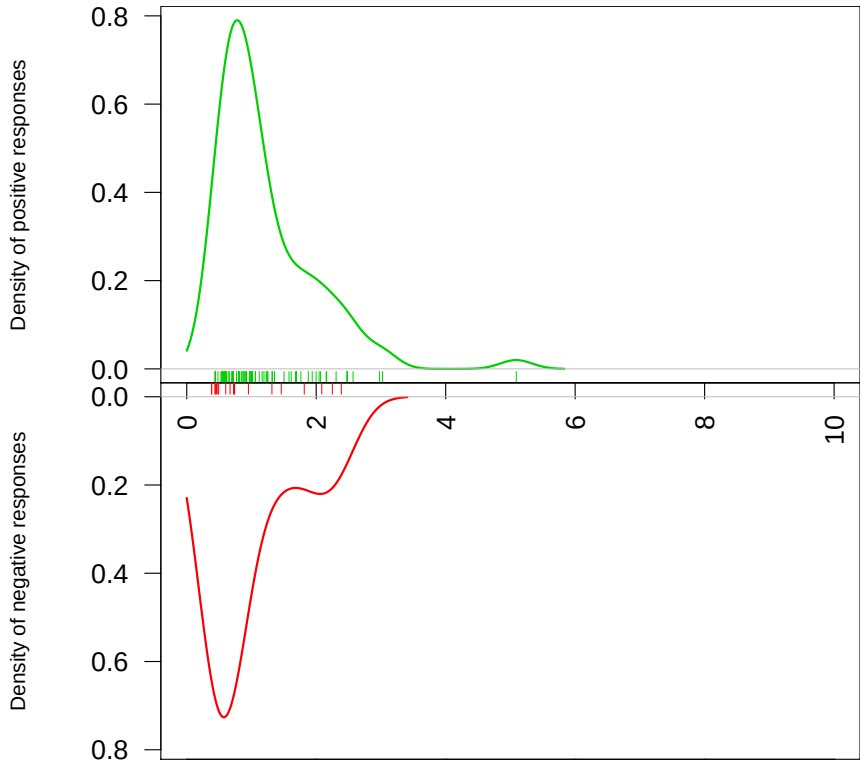

Figure 3: Plot drawn with `wiener_plot` function.

Again, the last five arguments of this function have the same meaning as for the two previously described functions. The first argument, however, is different: instead of a quantile, the first argument needs to be a probability, indicating the CDF-value to be looked up.

To round out the four standard distribution functions, the **RWiener** package further provides a function named `rwiener` that generates random values. Again, the last four input variables correspond to the model parameters and are the same as in the other distribution functions. The first argument `n` takes a number, indicating the desired number of randomly generated variates.

Plot function

To easily visualize data from the Wiener diffusion model, we can use the plot function `wiener_plot` as follows:

```
wiener_plot(dat)
```

This draws the two densities of the observed (or, in this case, generated) lower and upper responses. Figure 3 shows an example of this plot.

Parameter estimation with R's general-purpose optimization algorithms

In addition to the four distribution functions `dwiener`, `pwiener`, `qwiener`, `rwiener` and the plot function `wiener_plot`, the **RWiener** package provides four more functions designed for use in parameter estimation and model selection: (1) the `wiener_likelihood` function, which calculates the logarithm ℓ of the likelihood for given parameter values and data; (2) the `wiener_deviance` function, which calculates the model deviance -2ℓ; (3) the `wiener_aic` function, which calculates the model AIC (Akaike's Information Criterion) $-2\ell + 8$; and (4) the `wiener_bic` function, which calculates the model BIC (Bayesian Information Criterion) $-2\ell + 4\ln(N)$.

```
x <- c(2, .3, .5, .5)
wiener_likelihood(x=x, dat=dat)
wiener_deviance(x=x, dat=dat)
wiener_aic(x=x, dat=dat)
wiener_bic(x=x, dat=dat)
```

The first argument of these functions, `x`, is a vector containing the four parameter values of the model in the order: α, τ, β, δ. The second argument of these functions, `dat`, is a dataframe that contains the data points and the values of the two dependent variables: the RT and the boundary that was hit. This dataframe has to have the same shape as the dataframe `dat` as generated previously with the `rwiener` function. The first column of the dataframe has to contain the RTs and the second column of the dataframe has to contain the response type (*upper* vs. *lower*).

Estimating parameters for a single condition

In order to estimate model parameters for a given data set, one can use R's `optim` or `nlm` function:[2]

```
# using optim, first with Nelder-Mead algorithm, then with BFGS
optim1 <- optim(c(1, .1, .1, 1), wiener_deviance, dat=dat, method="Nelder-Mead")
optim2 <- optim(optim1[["par"]], wiener_deviance, dat=dat, method="BFGS", hessian=TRUE)

# using nlm, which uses a Newton-type algorithm
nlm1 <- nlm(p=c(1, .1, .1, 1), f=wiener_deviance, dat=dat)
```

The obtained model parameter estimates can then be used to draw further inferences or create predictions. The help pages of the functions presented here show additional information and examples of usage.

Estimating parameters for multiple conditions

The deviance function can be combined to compute a single loss value for multiple conditions. To do so, first create a new inline function:

```
many_drifts <- function(x, datlist) {
  l = 0
  for (c in 1:length(datlist)) {
    l = l + wiener_deviance(x[c(1, 2, 3, c+3)], datlist[[c]])
  }
  return(l)
}

# create a second data set and a list containing both data sets
dat2 <- rwiener(n=100, alpha=2, tau=.3, beta=.5, delta=1)
datlist <- list(dat, dat2)

# use nlm to estimate parameters
nlm1 <- nlm(p=c(1, .1, .1, 1, 1), f=many_drifts, dat=datlist)
```

If the input `x` is a parameter vector containing the first three parameters (α, τ, β) and then C drift rates, and `dat` is a list with C data sets, then the result will contain point estimates of three joint parameters in addition to C condition-specific drift rate estimates.

An alternative function can be defined that does not allow the drift rates to differ between conditions:

```
one_drift <- function(x, datlist) {
  l = 0
  for (c in 1:length(datlist)) {
    l = l + wiener_deviance(x, datlist[[c]])
  }
  return(l)
}

nlm2 <- nlm(p=c(1, .1, .1, 1), f=one_drift, dat=datlist)
```

Finally, the two competing models can be compared using the AIC criterion for model selection.[3]

```
AIC1 <- wiener_aic(x=nlm1$estimate, dat=datlist, loss=many_drifts)
AIC2 <- wiener_aic(x=nlm2$estimate, dat=datlist, loss=one_drift)
```

[2] Note that the `wiener_likelihood` function will return *Inf* for certain values, causing the `nlm` function to warn that the *Inf* was replaced by the maximum positive value.

[3] The input argument *loss*, with which a custom deviance function can be passed to compute the AIC for more complex models, is only available from **RWiener** v1.2.

For more details on the interpretation of the AIC criterion, see Wagenmakers and Farrell (2004).

Discussion

We presented **RWiener**, an R package that provides efficient algorithms to compute Wiener diffusion model functions and use these for further inference. This is the first package in R that provides full Wiener diffusion model functionality for R. We demonstrated how to use these functions and how to extend their usage to combine them with other R functions in order to do parameter estimation.

The presented functions can help researchers who work with 2CRT data to analyze their data in the Wiener diffusion model framework.

Authors' note

Correspondence concerning this article may be addressed to DW (dominik.wabersich@gmail.com) or JV (joachim@uci.edu). This project was supported by grant #1230118 from the National Science Foundation's Measurement, Methods, and Statistics panel to JV, and a travel grant from German Academic Exchange Service (PROMOS) to DW.

Bibliography

S. Blurton, M. Kesselmeier, and M. Gondan. Fast and accurate calculations for cumulative first-passage time distributions in Wiener diffusion models. *Journal of Mathematical Psychology*, 56(6):470–475, 2012. [p49, 55, 56]

D. J. Navarro and I. G. Fuss. Fast and accurate calculations for first-passage times in Wiener diffusion models. *Journal of Mathematical Psychology*, 53(4):222–230, 2009. [p49, 55]

F. Tuerlinckx, E. Maris, R. Ratcliff, and P. De Boeck. A comparison of four methods for simulating the diffusion process. *Behavior Research Methods*, 33(4):443–456, 2001. [p49]

J. Vandekerckhove. *Extensions and Applications of the Diffusion Model for Two-Choice Response Times*. PhD thesis, University of Leuven, 2009. [p50]

J. Vandekerckhove and F. Tuerlinckx. Fitting the Ratcliff diffusion model to experimental data. *Psychonomic Bulletin & Review*, 14(6):1011–1026, 2007. [p49]

J. Vandekerckhove and F. Tuerlinckx. Diffusion model analysis with MATLAB: A DMAT primer. *Behavior Research Methods*, 40(1):61–72, 2008. [p49]

J. Vandekerckhove, F. Tuerlinckx, and M. Lee. Hierarchical diffusion models for two-choice response times. *Psychological Methods*, 16(1):44, 2011. [p49]

A. Voss and J. Voss. fast-dm: A free program for efficient diffusion model analysis. *Behavior Research Methods*, 39(4):767–775, 2007. [p49]

E.-J. Wagenmakers. Methodological and empirical developments for the Ratcliff diffusion model of response times and accuracy. *European Journal of Cognitive Psychology*, 21(5):641–671, 2009. [p49]

E.-J. Wagenmakers and S. Farrell. AIC model selection using Akaike weights. *Psychonomic Bulletin & Review*, 11(1):192–196, 2004. [p54]

E.-J. Wagenmakers, H. van der Maas, and R. Grasman. An EZ-diffusion model for response time and accuracy. *Psychonomic Bulletin & Review*, 14(1):3–22, 2007. [p49]

T. V. Wiecki, I. Sofer, and M. J. Frank. HDDM: Hierarchical Bayesian estimation of the drift-diffusion model in Python. *Frontiers in Neuroinformatics*, 7(14), 2013. [p49]

Dominik Wabersich
Department of Cognitive Sciences
University of California, Irvine
Irvine, CA 92697
USA
dominik.wabersich@gmail.com

Joachim Vandekerckhove
Department of Cognitive Sciences
University of California, Irvine
Irvine, CA 92697
USA
joachim@uci.edu

Appendix

Mathematical descriptions of the density and distribution functions

Probability density function

The PDF of the diffusion model is that of the first-passage times of the accumulation process over the boundaries. That is, it is the expected distribution of the time until the process first hits or crosses one or the other boundary. This results in a bivariate distribution, over responses x and hitting times t. The PDF is approximated by Equation 1:

$$d\left(t, x=0|\alpha,\tau,\beta,\delta\right) = \frac{1}{\alpha^2}\exp\left[-\alpha\beta\delta - \frac{1}{2}\delta^2\left(t-\tau\right)\right] f\left(\frac{t-\tau}{\alpha^2}\middle|\beta\right). \tag{1}$$

The first passage time distribution for hits at the opposite boundary is $d\left(t, x=1|\alpha,\tau,\beta,\delta\right) = d\left(t, x=0|\alpha, 1-\beta,\tau,-\delta\right)$. In Equation 1, f can be either the *large-time representation*

$$f_L\left(u|\beta\right) = \pi\sum_{k=1}^{+\infty} k\exp\left(-\frac{k^2\pi^2 u}{2}\right)\sin\left(k\pi\beta\right)$$

or the *small-time representation*

$$f_S\left(u|\beta\right) = \frac{1}{\sqrt{2\pi u^3}}\sum_{k=-\infty}^{+\infty}\left(2k+\beta\right)\exp\left[-\frac{\left(2k+\beta\right)^2}{2u}\right]. \tag{2}$$

Either approximation of f can be more computationally efficient (in terms of the number of terms required to have the infinite sum converge below an error bound $\varepsilon = 10^{-10}$), depending on the parameter and data values. Specifically, Navarro and Fuss (2009) provide a decision rule to determine the more efficient formula: Equation 2 should be used if and only if

$$2 + \sqrt{-2u\log\left(2\varepsilon\sqrt{2\pi u}\right)} - \sqrt{\frac{-2\log\left(\pi u\varepsilon\right)}{\pi^2 u}} < 0,$$

where $u = \frac{t-\tau}{\alpha^2}$. The derivation of this decision rule, as well as demonstrations of its efficiency, are given in Navarro and Fuss (2009).

Cumulative distribution function

A similar rule for the efficient computation of the CDF can be found in Blurton et al. (2012)—there exists a large-time representation

$$\begin{aligned} p_L\left(t, x=0|\alpha,\tau,\beta,\delta\right) = P_0 - \frac{2\pi}{\alpha^2}\exp\left(-\alpha\beta\delta - \frac{\delta^2\left(t-\tau\right)}{2}\right) \\ \times\sum_{k=1}^{+\infty}\frac{k\sin\left(\pi k\beta\right)}{\delta^2+\left(k\pi/\alpha\right)^2}\exp\left[-\frac{1}{2}\left(\frac{k\pi}{\alpha}\right)^2\left(t-\tau\right)\right], \end{aligned} \tag{3}$$

and a small-time representation

$$\begin{aligned} p_S\left(t, x=0|\alpha,\tau,\beta,\delta\right) = P_0 - \operatorname{sgn}\delta\sum_{k=-\infty}^{+\infty}\left\{\exp\left(-2\delta\alpha k - 2\delta\alpha\beta\right)\,\Phi\left[\operatorname{sgn}\delta\frac{2\alpha k+\alpha\beta-\delta\left(t-\tau\right)}{\sqrt{t-\tau}}\right]\right. \\ \left. -\exp\left(2\delta\alpha k\right)\,\Phi\left[\operatorname{sgn}\delta\frac{-2\alpha k-\alpha\beta-\delta\left(t-\tau\right)}{\sqrt{t-\tau}}\right]\right\}, \end{aligned} \tag{4}$$

where sgn is the signum function and Φ denotes the cumulative normal distribution function. In both cases, P_0 is the probability of a hit at the lower boundary,

$$P_0 = \begin{cases} \frac{1-\exp[-2\delta\alpha(1-\beta)]}{\exp(2\delta\alpha\beta)-\exp[-2\delta\alpha(1-\beta)]} & \text{if } \delta \neq 0, \\ 1-\beta & \text{if } \delta = 0. \end{cases}$$

In order to reach an approximation with absolute error not exceeding $\varepsilon = 10^{-10}$, the infinite sums in Equations 3 and 4 are approximated with K_L and K_S terms, respectively (derivation of these limits is given in Blurton et al., 2012). K_L and K_S are the smallest integers that satisfy the following constraints:

$$\begin{cases} K_L^2 & \geq & \frac{1}{t-\tau}\left(\frac{\alpha}{\pi}\right)^2 \\ K_L^2 & \geq & -\frac{2}{t-\tau}\left(\frac{\alpha}{\pi}\right)^2 \left\{\log\left[\frac{1}{2}\varepsilon\pi\left(t-\tau\right)\left(\delta^2+\frac{\pi^2}{\alpha^2}\right)\right]+\delta\alpha\beta+\frac{1}{2}\delta^2\left(t-\tau\right)\right\} \end{cases}$$

and

$$\begin{cases} K_S & \geq & \beta-1+\frac{1}{2\delta\alpha}\log\left\{\frac{\varepsilon}{2}\left[1-\exp\left(2\delta\alpha\right)\right]\right\} \\ K_S & \geq & \frac{1}{2\alpha}\left[0.535\sqrt{2\left(t-\tau\right)}+\delta\left(t-\tau\right)+\alpha\beta\right] \\ K_S & \geq & \frac{1}{2}\beta-\frac{1}{2\alpha}\sqrt{t-\tau}\;\;\Phi^{-1}\left\{\frac{\varepsilon\alpha}{0.3\sqrt{2\pi(t-\tau)}}\exp\left[\frac{1}{2}\delta^2\left(t-\tau\right)+\delta\alpha\beta\right]\right\} \end{cases}.$$

Due to the larger computational effort of the small-time representation of the CDF, the large-time representation is selected if $K_L/K_S < 10$.