Computational Statistics (2022) 37:1653–1693
https://doi.org/10.1007/s00180-021-01175-3

ORIGINAL PAPER

`Markovchart`: an `R` package for cost-optimal patient monitoring and treatment using control charts

Balázs Dobi[1] · András Zempléni[1]

Received: 23 May 2021 / Accepted: 10 November 2021 / Published online: 27 December 2021

Abstract
Control charts originate from industrial statistics, but are constantly seeing new areas of application, for example in health care (Thor et al. in BMJ Qual Saf 16(5):387–399, 2007. https://doi.org/10.1136/qshc.2006.022194; Suman and Prajapati in Int J Metrol Qual Eng, 2018. https://doi.org/10.1051/ijmqe/2018003). This paper is about the `Markovchart` package, an `R` implementation of generalised Markov chain-based control charts with health care applications in mind and with a focus on cost-effectiveness. The methods are based on Zempléni et al. (Appl Stoch Model Bus Ind 20(3):185–200, 2004. https://doi.org/10.1002/asmb.521), Dobi and Zempléni (Qual Reliab Eng Int 35(5):1379–1395, 2019a. https://doi.org/10.1002/qre.2518, Ann Univ Sci Budapestinensis Rolando Eötvös Nomin Sect Comput 49:129–146, 2019b). The implemented ideas in the package were motivated by problems encountered by health care professionals and biostatisticians when assessing the effects and costs of different monitoring schemes and therapeutic regimens. However, the implemented generalisations may be useful in other (e.g., engineering) applications too, as they mainly revolve around the loosening of assumptions seen in traditional control chart theory. The `Markovchart` package is able to model processes with random shift sizes (i.e., the degradation of the patient's health), random repair (i.e., treatment) and random time between samplings (i.e., visits) as well. The article highlights the flexibility of the methods through the modelling of different disease progression and treatment scenarios and also through an application on real-world data of diabetic patients.

Keywords Control chart · Cost-effectiveness · Markov chain · Health care · `R`

✉ Balázs Dobi
dobibalazs@inf.elte.hu

András Zempléni
andras.zempleni@ttk.elte.hu

[1] Department of Probability Theory and Statistics, Faculty of Sciences, Eötvös Loránd University, 1/C Pázmány Péter sétány, Budapest 1117, Hungary

1 Introduction

Since their first applications in the 1920s, control charts have proliferated to many areas, and several of these are outside of the traditional industrial settings (Montgomery 2009; Zhou and Zhang 2015; Sales et al. 2016; Brooks et al. 2013). One of the less-traditional areas is health care where many different types of control chart applications have been introduced (Thor et al. 2007; Suman and Prajapati 2018). Several applications focus on quality control such as mortality rate, length of hospital stay, number of complications, etc. Control charts have also been applied for the monitoring of the state of patients, see e.g., Correia et al. (2011). The `Markovchart` `R` (R Core Team 2020) package deals with this kind of control, namely problems emerging from the monitoring of a single patient at a time.

Often, the goal of control charts (and generally of statistical process control) is to improve a process from a certain point of view e.g., minimise the number of faulty products. There are different approaches to the measuring of control chart performance, one of which is statistical optimality. This approach aims to maximise the in-control (IC) average run length (ARL) and minimise the out-of-control (OOC) ARL, which is the average time until an alarm signal, see Montgomery (2009) for further read on the subject. Another approach is to define costs to various parts of the control chart model and try to optimise the setup by minimising the total cost. One of, if not the most popular cost-efficient model is Duncan's cycle model (Duncan 1956).

There are many different statistical process control models and implementations across various platforms. The most popular control charts and their visualisation methods can be found in `R` (R Core Team 2020), `SAS` (SAS Institute Inc. 2013), `Stata` (StataCorp LLC 2019), `SPSS` (IBM Corporation 2019), `Excel` (Microsoft 2019; Buttrey 2009) etc. Statistical-process-control-related packages in `R` include `spc` (Knoth 2020), `edcc` (Zhu and Park 2013b), `qicharts` (Anhoej and Roeder 2017), `qcc` (Scrucca et al. 2017), `qcr` (Flores et al. 2020), `ggQC` (Grey 2018) and possibly other, more elusive ones.

In `R`, only the `edcc` package (Zhu and Park 2013a) has methods for cost-optimisation to the authors' knowledge. The package essentially implements Duncan's cycle model for multiple chart types, carries out optimisation with respect to the expected cost per hour and provides visualisation. The free parameters in the package are the sample size, sampling interval and the critical value. The `Markovchart` package also provides many of the same functionality, but based on different, more general methodology with focus on health care applications.

The `Markovchart` package implements a Markov-chain-based cost-optimisation framework developed originally by Zempléni et al. (2004). It is similar to Duncan's model in a sense that it also partitions the process into different states. The advantage of the framework is that it allows for extensions that are required in the monitoring of several process types. Random shift sizes were described already in Zempléni et al. (2004). Random repair effectiveness and random sampling time were introduced in Dobi and Zempléni (2019a), and different shift size distributions in Dobi and Zempléni (2019b). These generalisations are all necessary for proper modelling of certain processes, especially in health care. It can be noted though, that similar processes and monitoring problems also arise in many areas, e.g., Zempléni et al. (2004) applied the method for pulp production for a Portuguese pulp plant. As the package was made available at the Comprehensive R Archive Network (Dobi and Zempléni 2020), users from many areas can use the framework in their own setting of application.

The additional features in the Markovchart package come at a price though: the implemented optimisation uses only two free parameters, the sampling interval and the critical value. There is no theoretical obstacle for optimisation with respect to the sample size or any other parameters, in fact the article of Zempléni et al. (2004) shows examples for optimisation using the sample size, and in this article we compare repair mechanisms (i.e., therapies) in Sect. 5. The problem with the implementation of sample sizes greater than 1 is that it makes practical implementation and calculation a serious hurdle. Nonetheless, this is an inconsequential problem in most cases, as the goal of the Markovchart package is the monitoring of a single patient for a time period.

The Markovchart package features three main functions: Markovstat for process behaviour estimation, i.e., stationary distribution calculation. This function has highly customisable parts which provide a general framework for many processes and monitoring environments. Markovchart is the main function of the package and is used for cost calculation and optimisation. It is possible to plot the results of this main function as a function of the free parameters using a plotting method. The third function, Markovsim is for simulating processes with the assumptions used by the control chart models.

The rest of the article is organised in the following way: Sect. 2 describes the mathematical model used in the R package Markovchart. Section 3 documents the implemented methods and functions with examples. Section 4 compares different model results, including comparison to the edcc package. In Sect. 5, we show an example application based on the real-world data of diabetic patients. Section 6 concludes the article.

2 Mathematical background

The description of the Markov chain-based framework below is just a brief introduction and summary necessary for understanding the mechanics of the Markovchart package. For further reading and more detailed descriptions see Zempléni et al. (2004), Dobi and Zempléni (2019a, b).

2.1 Notions and notations

In many patient monitoring situations we are only interested in one-sided deteriorations e.g., high blood pressure, high blood lipid level, high blood sugar level, etc. The opposite of these examples (low levels) can also be harmful to the patient, but those symptoms usually fall under different diseases, thus can be treated as separate but still one-sided monitoring problems. Patient state deterioration in the process control sense is modelled here as a shift in the expected value of the monitored characteristic. Because of these considerations, we shall assume that only positive shift occurs in the expectation of the characteristic in case of no alarm. This model corresponds to an X-chart setup with sample size $n = 1$, time between samplings (sampling interval) h and one-sided critical value k. There are two free parameters which are in the focus of optimisation: k and h.

k and h are free parameters, but the following values are supposed to be known and other aspects of the model are also assumed [many of these assumptions also appear in general control chart theory, see e.g., Montgomery (2009) and Mortarino (2010)]:

- The distribution of the measurement error is known with expectation 0 and standard deviation σ.
- The shift intensity ($1/s$ = the inverse of the expected number of shifts in a unit time) is constant and assumed to be known.
- The shift size distribution and its parameters are assumed to be known.
- The process does not repair itself. The distribution describing the repair size (effectiveness) and its parameters are assumed to be known.
- The sampling might not take place or may be unsuccessful. The distribution describing the sampling probability and its parameters are assumed to be known.

The above point may require some elaboration: The time between shifts are usually assumed to be exponentially distributed, we will also use this principle. Also, let the cumulative distribution function (CDF) of the measurement error be denoted by ϕ. During implementation we will assume that the IC process distribution is normal with parameters μ_0 (target value/IC expectation) and σ (process standard deviation). The OOC process distribution is consequently also normal with shifted μ OOC expectation and the same σ standard deviation, as shifts are assumed to only change the expected value. Regarding repair, as the process does not repair itself the repair is initiated by an alarm and it is treated as an instantaneous event, thus all costs related to repairing should be included in the repair cost. For example if a major repair entails higher cost, then this should also be reflected in the calculation. Also, the repair may not be perfect, in this case the process remains at a strictly higher expected value than the μ_0 target value, even after repair. At least part of the repair cost may also occur during OOC operation. This is necessary because in many medical applications treatment is continually present (e.g., medicine has to be taken daily). The time between shifts, the shift sizes and repair effectiveness are all assumed to be independent from each other. The costs only depend on the distance from the target value μ_0, thus repair effectiveness and costs are only connected through the initial state before repair (i.e., a successful repair does not incur higher costs than an unsuccessful).

Using the above assumptions, the future distances from μ_0 are only dependent on the current distance. This way, one can define a Markov chain. The states of this Markov chain are defined at the sampling times and the states can be identified by the measured value and the actual (unobservable) background process, namely whether there was a shift from the target value in the parameter and whether there was an alarm. The difference between the measured value and the unobserved background process is due to the process standard deviation. So we can define four basic state types:

- No shift—no alarm: in-control (IC)
- Shift—no alarm: out-of-control (OOC)
- No shift—alarm: false alarm (FA)

– Shift—alarm: true alarm (TA)

The above assumptions leave ample room for parameters to be estimated for practical use. See Sect. 5 for examples.

2.2 Generalised model

As mentioned in Sect. 1, our approach contains three important differences compared to the traditional economic control chart models. First is the random shift size, introduced for the cost-optimal approach by Zempléni et al. (2004). The second is the random repair size and the third is the random sampling time, both introduced in Dobi and Zempléni (2019a). Moreover, Dobi and Zempléni (2019b) introduced the possibility of different shift size distributions, which is also implemented in the Markovchart package. In the next paragraphs the mathematical definition of the processes and distributions involved in the Markov chain-based cost-optimal control chart models are given.

Let τ_i denote the random shift times on the real line and let ρ_i be the shift size at time τ_i. Assume that all ρ_i are non-negative and independent random variables, and that the shift sizes are independent from τ_i as well. Let the number of shifts in the time interval $(0; t)$ be denoted by ν_t. ν_t has a discrete distribution with support over $\mathbb{N}_0$. The resulting random process Z_t has step functions as trajectories, which are monotonically increasing. Thus the CDF of the actual (unobservable) process values at a given time t from the start (where it was in the in-control state) can be written the following way (assuming that there was no alarm before t):

$$Z_t(x) = \begin{cases} 0 & \text{if } x < 0, \\ \nu_t(0) + \sum_{k=1}^{\infty} \nu_t(k)\Psi_k(x) & \text{if } x \geq 0, \end{cases} \tag{1}$$

where Ψ_k is the CDF of the k-fold convolution of shift sizes ρ_i. The case $x = 0$ means there was no shift till t (its probability is $\nu_t(0)$). $\Psi_k(0) = 0$ since only positive shifts can occur.

Currently two shift size distributions are implemented in the Markovchart package: exponential and exponential-geometric mixture shifts. In the case of the first one Ψ_k simply becomes the gamma (Erlang) distribution as it is a sum of k independent, identical exponentially distributed random variables. The second one has a more complicated form as the convolution of the negative binomial (sum of the geometric shifts) and the gamma (sum of the exponential shifts) distribution. For further reading see Dobi and Zempléni (2019b). ν_t is implemented in the package as the Poisson distribution with parameter st (the expected number of shifts per unit time multiplied by the length of the interval), as we assume that τ_i depends on neither the previous shifts nor the current process value, so the shift times form a homogeneous Poisson process.

The next generalisation is imperfect repair, which means that the treatment will not have perfect results on the health of the patient, or—in industrial settings—that the original condition of the machines cannot be fully restored. In this case, the imperfectly repaired states act as OOC states. It is assumed that the repair cannot worsen the state of the process, but the operation of an imperfectly repaired process will still cost the same as an equally shifted OOC process, thus 'repaired' and 'regular' OOC states do not need distinction during the cost calculation. The imperfect repair has to fit into the Markov chain-based framework, thus it is assumed that alarms, shifts and repair effectiveness are all independent. Let the random variable R determine the proportion of the remaining distance from μ_0 after repair. In the `R` implementation we assume that R has a $Beta(\alpha, \beta)$ distribution with known parameters (see Sect. 3.1 for more details).

Yet another generalisation is the random sampling time. In certain environments, the occurrence of the sampling at the prescribed time is not always guaranteed. For example in health care, the patient or employee compliance can have significant effect on the monitoring, thus it is important to take this into account during the modelling too. Here it is modelled in a way, that the sampling is not guaranteed to take place—e.g., the patient may not show up for control visit. This means that the sampling can only occur according to the sampling intervals, for example at every nth days, but is not a guaranteed event. One can use different approaches when modelling the sampling probability, the `Markovchart` package implements two cases. The first one assumes that too frequent samplings will decrease compliance. This assumption is intended to simulate the situation in which too frequent samplings may cause increased difficulty for the staff or the patient—leading to decreased compliance. The probability of a successful sampling as a function of the prescribed time between samplings is modelled in this case using a logistic function:

$$T_h = \frac{1}{1 + e^{-q(h-z)}},$$

where $q > 0$ is the steepness of the curve, $z \in \mathbb{R}$ is the value of the sigmoid's midpoint and h is the time between samplings.

In the other implemented approach it is assumed that too frequent samplings will decrease compliance and increased distance from the target value will increase compliance. This assumption means that a heavily deteriorated process or health state will ensure a higher compliance. The probability of a successful sampling as a function of the prescribed time between samplings and the distance from the target value is modelled using a beta distribution function:

$$T_h^*(v) = P\left(W_h < \frac{v}{V}\right), \tag{2}$$

where W_h is a $Beta(\mathrm{a}/h, \mathrm{b})$ distributed random variable (representing the aversion from sampling), v is the distance from the target value, h is the time between samplings, V is the maximum distance from μ_0 taken into account (thus $0 < v < V$ is assumed; this notation will be used throughout the paper). V here is effectively the distance where we expect maximal possible compliance.

Example curves for the successful sampling probability can be seen in Fig. 1. It shows that longer time between samplings and greater distances from the target value increase the probability of successful sampling.

A process with the above assumptions and generalisations is visualised in Fig. 2.

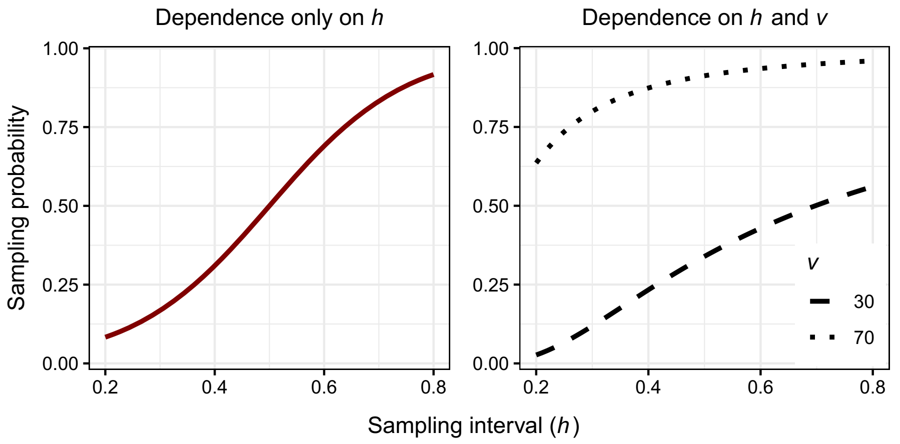

Fig. 1 Sampling probabilities for $q = 8$, $z = 0.5$ on the left, and for $a = 1$, $b = 3$, $V = 100$ on the right

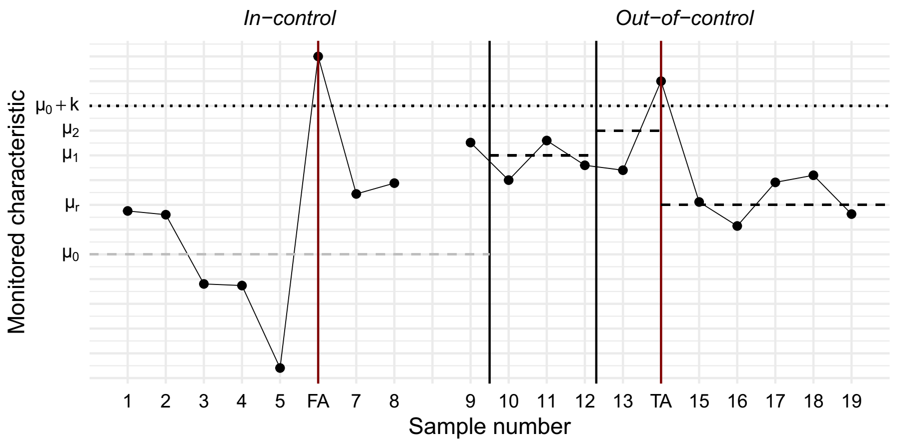

Fig. 2 Definition of states. Dashed line: expected value, Black vertical line: shift in the expected value, Dotted line: critical value, FA: False alarm, TA: True alarm

One can see that the process starts from an IC state at μ_0. Even though there is no shift, an alarm signal is still possible, which is a FA. After some time, there may be multiple shifts (to μ_1 and later to μ_2) in the value of the monitored characteristic (i.e., expected value), which create OOC states. During this phase an alarm is called a TA, which induces a repair which is not perfect, thus the process stays OOC, but at a level (μ_r) which is closer to μ_0 than before the repair. It is possible that the sampling will not take place, as seen between the 8th and 9th samplings.

2.3 Transition matrix and stationary distribution

For cost optimisation purposes we would like to find a discrete Markov chain that describes the patient's state. The stationary distribution of this chain essentially approximates the distribution of the monitored characteristic at the time of samplings. This requires the discretisation of the above defined functions, which in turn will allow us to construct a discrete time Markov chain with discrete state space. For technical details on this discretisation see Sect. 3.1.

The size and composition of the transition matrix—let us denote it with $\boldsymbol{\Pi}$—depends on whether or not we are considering random shift sizes. The traditional model which is similar to Duncan's cycle model has a 4×4 transition matrix corresponding to the 4 possible states described in Sect. 2.1. Transition probabilities can be written in this case using the ϕ process distribution (including measurement error) and the cumulative distribution function of the exponential distribution (time between shifts). Using the generalised model, the transition probabilities can be written using the ϕ process distribution, the Z_t shift size distribution (see (1)), the $Beta(\alpha, \beta)$ repair size distribution and the T_h (or T_h^*, see (2)) sampling probability. Note that in every scenario the states are defined at the time of samplings before repair and only positive shifts are possible between samplings. It is not necessary to use all generalisations at once: it is possible e.g., to use perfect sampling, while keeping the random shift size and the random repair. (There is a restriction though, namely that both the random repair and the random sampling time are only implemented for random shift sizes.) Let V_d be the number of considered distances - discretised shift sizes within the V-length interval. The size of the transition matrix is $2V_d \times 2V_d$ in the generalised case since every shift size has two states: one with and one without alarm. We assume that the first V_d columns are states without alarm, the second V_d are states with alarm. Once the process leaves the in-control state it will only return, if the repair is perfect. This is due to the nature of the imperfect repair we have discussed above. Whether or not the repair is perfect depends on the assumptions, this is parametrisable in the package. For a more detailed description of the transition matrix and the transition probabilities see Dobi and Zempléni (2019a).

In the perfect repair case the transition matrix $\boldsymbol{\Pi}$ defines a Markov chain with a discrete, finite state space with one positive recurrent class. If the repair is imperfect, the transition matrix has one transient, inessential class (the IC and the FA state) and one positive recurrent class (OOC and TA states). In finite Markov chains, the process leaves a transient class with probability one. The problem of finding the stationary distribution of the Markov chain is thus reduced to finding a stationary distribution within the recurrent class of the chain. Since there is a single positive recurrent class which is also aperiodic, we can apply the Perron–Frobenius theorem to find the stationary distribution (Meyer 2000). Consider now $\boldsymbol{\Pi}$ for only the positive recurrent class (this is the only class if the repair is perfect), let us denote it by $\boldsymbol{\Pi}^*$. The stationary distribution—which is the left eigenvector of $\boldsymbol{\Pi^*}$, normalised to sum to one—is unique and exists with strictly positive elements. Finding the stationary distribution is then reduced to solving the following equation: $\boldsymbol{\Pi}^{*T}\mathbf{f_0} = \mathbf{f_0}$, i.e., f_0 is the left eigenvector of $\boldsymbol{\Pi}^*$. This amounts to solving as many equations as the number of states in the recurrent class for the same number of variables. In the setup described above there is always a solution since the transition matrix has a unique largest real eigenvalue and the corresponding eigenvector can be chosen to have strictly positive components. The stationary distribution is then:

$$\mathbf{P} = \frac{\mathbf{f_0}}{\sum_{i=1}^{N} \mathbf{f_{0_i}}},$$

where N is the number of states within the recurrent class of the Markov chain which (depending on the generalisations used) can be 4, $2V_d$ or $2V_d - 2$.

2.4 Cost function

The authors have documented (Zempléni et al. 2004; Dobi and Zempléni 2019a) general-purpose but fixed-form cost functions to be used in the Markov chain-based control chart model. On the other hand, the `Markovchart` package implements a highly parametrisable cost function. The general form of the expected cost function per unit time is

$$\mathrm{E}(C) = \mathbf{c} \cdot \mathbf{P}, \tag{3}$$

where $\mathrm{E}(C)$ is the expected cost per unit time, $\mathbf{c}$ is a vector of the total cost per unit time associated with each state and $\mathbf{P}$ is the stationary distribution (vector of probabilities), defined above. It is the expected value of the discrete distribution defined by the costs and the stationary distribution of the Markov chain. As the calculation of the $\mathbf{c}$ vector is highly customisable in the `Markovchart` package, here we will mainly show the default functions.

In the classical fixed shift size case, the $\mathbf{c}$ vector is

$$\mathbf{c} = \left\{ \frac{c_s}{h}, \frac{c_s + c_f}{h}, \frac{c_s}{h} + c_o, \frac{c_r + c_s}{h} + c_o B_h \right\}, \tag{4}$$

so the coordinates correspond to the IC, FA, OOC and TA states respectively. c_s is the sampling cost, c_f is the FA cost, c_r is the TA (repair) cost and c_o is the OOC operation cost per unit time. B_h is the fraction of time between consecutive observations where the shift occurred and remained undetected (Zempléni et al. 2004; Mortarino 2010). The sampling and repair costs need to be divided by the length of the sampling interval in order to get the cost of a unit time of operation. (It can be noted though that this is parametrisable for the repair cost in the package.) The OOC cost should be given by default as per unit time, so it has not to be divided.

In the random shift size case, c_o and c_r need to be defined for each distance from the target value. The relationship between the distance from the target value and the resulting OOC cost may be non-trivial. Many potential models could be used, in the `Markovchart` function the calculation is based on a Taguchi-type loss function. This means that a squared loss function is assumed. The default cost function of the OOC cost is:

$$c_o(v) = c_{ob} + c_{os}\mathcal{A}_h^2(v),$$

where c_{ob} is the base OOC cost per unit time, whereas c_{os} is the distance-scaling OOC cost per unit time. v is the distance from the target value and $\mathcal{A}_h^2(v)$ is the expected total squared distance from this starting value on the condition that the sampling interval is h. This $\mathcal{A}_h^2(v)$ incorporates the distances (the base of the losses) incurred not just at the time of the sampling, but also between samplings—hence the dependency on h: if the process operates for long times without check and thus without alarms and repairs, it is expected that it will be more deteriorated with respect the squared average. The distances are squared as we use a Taguchi-type loss function. The calculation is more complicated than the one for B_h at Eq. 4, because the shift size is not fixed here, so an expected value needs to be computed. The calculation is based on the expected behaviour of the process between the current sampling and the next one, for more details on $\mathcal{A}_h^2(v)$ see Dobi and Zempléni (2019a, b). Even if the user defines a custom cost function for the OOC cost this $\mathcal{A}_h^2(v)$ term must be included. A closed form solution has been developed and implemented for the calculation of the total expected squared distances, considerably decreasing the running times (Dobi and Zempléni 2021). For details, see the "Appendix".

The default function for the repair cost in the random shift size case is a linear function:

$$c_r(v) = c_{rb} + c_{rs}v,$$

where c_{rb} is the base TA (repair) cost and c_{rs} is the distance scaling TA cost. Since there is no fixed squared term, a custom function can be defined more freely here.

Thus the total cost per unit time associated with an alarm state v distance from the target value is

$$c_{v,TA}(v) = \frac{c_s}{h}T_h(v) + c_o(v) + \frac{c_r(v)}{h},$$

and with an OOC state is

$$c_{v,OOC}(v) = \frac{c_s}{h}T_h(v) + c_o(v) + \frac{p_r c_r(v)}{h}. \tag{5}$$

p_r is the amount (proportion) of repair cost occurring during OOC operation. As we mentioned before, this is necessary because in many medical applications treatment is continually present. In classical, industrial settings p_r would be 0 in most applications. Again, the sampling and repair costs need to be divided by the length of the sampling interval and the OOC cost should be given by default as per unit time. Whether or not the repair cost is given per unit time or as the total cost of treatment is optional and is parametrisable in the package. The cost of sampling needs to be weighted by the probability of the sampling ($T_h(v)$). This probability is incorporated into the stationary distribution—this is why it is not needed for the repair cost in the equation above. If the sampling is not random then $T_h(v) \equiv 1$.

In certain fields of application, the reduction of the cost standard deviation can be just as or even more important than the minimisation of the expected cost, see e.g., McCracken and Chakraborti (2013). It is usually straightforward to calculate not just the expected value, but further moments and the variance as well. In the classical industrial setting with fixed shift size, it can be assumed that the process is the only source of the variance, as it is expected that the same OOC and repair cost will be incurred every time if the same problem (shift size) occurs. This of course can be different in certain fields and applications, but for the simplicity of the most basic model we will not assume further sources of cost variance here, which is then simply $\mathrm{Var}(C)$.

However, the generalised model has random shift sizes with possibly random repair. Also, in health care settings it cannot be assumed that a disease level (distance from μ_0) will incur the same costs each time. Thus it is necessary to estimate not just costs but their variances as well. In this case, the total variance can be calculated using the law of total variance, i.e., using the cost variance due to the process ($\mathrm{Var}(C)$) and the weighted sum of the additional variances associated with each state:

$$\mathrm{Var}_{tot}(C) = \boldsymbol{\varsigma}_{\mathbf{tot}} \cdot \mathbf{P} + \mathrm{Var}(C), \tag{6}$$

where ς_{tot} is the vector of the total variances associated with each state. For one element (a v distance) this can be written as $\varsigma_{tot}(v) = \varsigma_o(v) + \varsigma_r(v)$, i.e., the sum of the OOC and repair cost variance components. The sampling cost is constant, thus does not add variance to the system. The default functions for cost variance estimation have the same form and restrictions as above for the costs. Note, that for fixed shift size $\mathrm{Var}_{tot}(C) = \mathrm{Var}(C)$ in the Markovchart package. Of course there could be e.g., random costs associated with fixed shifts but we did not (yet) explore this model in our package.

During optimisation one can consider a linear combination of the expected cost and the cost standard deviation:

$$G = p\mathrm{E}(C) + (1 - p)\mathrm{sd}_{tot}(C). \tag{7}$$

G is the value to be minimised, p is the weight of the expected cost ($0 \le p \le 1$) and $\mathrm{sd}_{tot}(C) = \sqrt{\mathrm{Var}_{tot}(C)}$. The standard deviation is preferred over the variance during model interpretation as it keeps the unit of measurement.

3 Implementation

3.1 Discretisation details

In our approach a vector of probabilities is needed to represent the stationary distribution due to the discrete state space time-homogeneous Markov chain. This means that many aspects of the continuous model need to be discretised, including the shift size PMF. Discretisation may introduce a bias: in reality, the distance from the target value can fall anywhere within a discretised interval, which has to be represented by a single value—for this the midpoint of the interval is used.

Let Δ be the unit of the discretisation (length of intervals) and V_d the number of states created by discretisation. The IC state is denoted by 0, thus the upper endpoints of the intervals are $0, \dots, (V_d - 1)\Delta$. We define a function for notational convenience, this will be used for reducing the discretisation bias:

$$\Delta_+(u) = u\Delta + \frac{\Delta}{2}, \quad u = 0, \dots, V_d - 1,$$

$u\Delta$ would simply be the lower boundary of the interval in consideration. The $\frac{\Delta}{2}$ term is added to get the middle of the interval.

We can define the discretised shift size PMF for a t long sampling interval, given that the starting state is i and the size of the shift (measured in discretised units) is j:

$$z_{t,i}(j) = \begin{cases} \nu_t(0) & \text{if } j = 0, i = 0, \\ \sum_{k=1}^{\infty} \nu_t(k)\big(\Psi_k(j\Delta) - \Psi_k((j-1)\Delta)\big) & \text{if } 1 \le j \le V_d - 1, i = 0, \\ \nu_t(0) + \sum_{k=1}^{\infty} \nu_t(k)\Psi_k(\Delta_+(0)) & \text{if } j = 0, i \neq 0, \\ \sum_{k=1}^{\infty} \nu_t(k)\big(\Psi_k(\Delta_+(j)) - \Psi_k(\Delta_+(j-1))\big) & \text{if } 1 \le j \le V_d - i - 1, i \neq 0, \end{cases} \tag{8}$$

where Ψ_k is the CDF of the sum of k independent, identically distributed ρ_i shift sizes. For $j = 0$ the function is the probability of staying at the current level. The $i = 0$ case represents shifts from the healthy, IC state. This case requires special treatment, since this value is exactly given, unlike the other cases, where the value can fall anywhere within the discretised interval (and is represented by the midpoint of the interval). Naturally, the infinite sums can only be approximated during application. This discretisation scheme works for continuous, discrete and mixture distributed shifts as well.

The discretised version of the repair size distribution can be written the following way:

$$R(l, m) = P\left(\frac{m}{l + 1/2} \le r < \frac{m+1}{l + 1/2}\right),$$

where r is a $Beta(\alpha, \beta)$ distributed random variable. l is the number of discretised distances closer to μ_0 than the current one. m is the index for the repair size interval we are interested in ($m \le l$), with $m = 0$ meaning the best possible repair. The repair is assumed to move the expected value towards the target value by a random percentage, governed by r. Even though discretisation is required for practical use of the framework, in reality the repair size distribution is continuous. To reflect this continuity in the background, the probability of perfect repair is 0 ($m = 0$ corresponds to cases in the nearest interval to the target). l is set to be 0 if there is nothing to be repaired, thus $R(0, 0) = 1$. The $1/2$ terms are necessary because of the discretisation, i.e., because the actual value in the interval is approximated by the midpoint.

In case when the sampling probability also depends on the shift size,

$$T_h(i) = P\left(W_h < \frac{i + 1/2}{V_d}\right), \; i = 0, \ldots V_d - 1,$$

where W_h is the *beta* distributed random variable introduced in Eq. 2, i is the ith state's distance from the target value in discretised units as in Eq. 8 (note that the smallest discretised distance is 0 and thus the greatest is $V_d - 1$). V_d is again the number of considered intervals. The $+1/2$ term is necessary because we use the midpoint of an interval due to discretisation. The probability is defined in a way to avoid 0 or 1 sampling probability, as the patient behaviour is not assumed to be deterministic. Other distributions may be considered for modelling but the *beta* distribution allows for a wide range of shapes, thus many different patient behaviours can be modelled this way.

3.2 Function `Markovstat`

The `Markovstat` package features three main functions. The print method will also be highlighted in further subsections. These are all connected to the `Markovchart` function:

- `Markovstat` for process behaviour, i.e., stationary distribution.
- `Markovchart` for cost calculation with optimisation.
- `plot.Markov_grid` for plotting the results of `Markovchart` as the function of the free parameters (h and k).
- `Markovsim`, a function for simulating processes handled by `Markovchart`.

Work with the package usually starts with the `Markovstat` function, as the first parameter of the `Markovchart` function is created by it. The description of the most important parameters can be seen below:

- `shiftfun`: A string defining the shift size distribution to be used. Must be either `'exp'` (exponential), `'exp-geo'` (exponential-geometric mixture) or `'deg'` (degenerate). Use `'deg'` for fixed shift size with perfect repair and guaranteed sampling, i.e., Duncan's traditional cycle model.
- `h`: The time between samplings. Must be a positive value.
- `k`: The control limit (critical value). Must be a positive value. Only one sided shifts are allowed, thus there is only one control limit.
- `sigma`: Process standard deviation (the distribution is assumed to be normal).
- `s`: Expected number of shifts in an unit time interval.
- `delta`: Expected shift size. Used as the parameter of the exponential distribution (`shiftfun = 'exp'` or `'exp-geo'`), or simply as the size of the shift (`shiftfun = 'deg'`).
- `probmix`: The weight of the geometric distribution in case of exponential-geometric mixture shift distribution; should be between 0 and 1.
- `probnbin`: The probability parameter of the geometric distribution in case of exponential-geometric mixture shift distribution; should be between 0 and 1.

- RanRep: Logical. Should the repair be random? Default is FALSE (the repair is perfect, the process is always repaired to the target value). The repair is always perfect (non-random) for shiftfun = 'deg'.
- alpha: First shape parameter for the random repair beta distribution.
- beta: Second shape parameter for the random repair beta distribution.
- RanSam: Logical. Should the sampling be random? Default is FALSE (no). The sampling is never random for shiftfun = 'deg'.
- StateDep: Logical. Should the sampling probability also depend on the distance from the target value (state dependency)? (If TRUE, a beta distribution is used for the sampling probability, if FALSE then a logistic function.)
- a: First parameter · h for the random sampling time beta distribution. The first shape parameter is a/h to create dependency on the time between samplings as described at the StateDep parameter.
- b: Second shape parameter for the random sampling time beta distribution.
- Vd: Integer discretisation parameter: the number of states in the equidistant discretisation of the state space.
- V: Numeric discretisation parameter: the maximum (positive) distance from the target value taken into account.

The type of models and features allowed depend foremost on the type of shift size distribution used. This can be degenerate (fixed shift size), exponential and mixture (exponential-geometric). Random shift size, random repair and random sampling are tied to non-degenerate distributions.

The value of the function is a Markov_stationary object, which is a list of length 3, detailing the properties of the stationary distribution and the transition matrix.

- Stationary_distribution: Stationary distribution of the Markov chain. The probabilities in the stationary distribution are labelled. If shiftfun is 'deg' then the stationary distribution is always of length 4. In this case the out-of-control and true alarm states are at a distance delta from the target value, and the in-control and the false alarm state are always at the target value. If shiftfun is not 'deg' then there are multiple out-of-control and true alarm states. In this case the stationary distribution of the Markov chain is a named numeric vector of length Vd*2. The length is double of Vd because each state has an alarm and a non-alarm (OOC) version. There is one IC and one FA state but there are multiple OOC and TA states (for all discretised distances). The label contains the following information separated by the '_' character: state type, state index number, distance from μ_0 represented by the state (the midpoint of the interval).
- Transition_matrix: The transition matrix of the Markov chain. Not printed.
- Param_list: Parameters given to the function and various technical results used by the Markovchart function. Not printed.

The simplest case is the same as Duncan's cycle model on an $\bar{X}$ chart with a sample size of 1. An example can be seen below:

```
> stat_simple <- Markovstat(shiftfun = 'deg', h = 1, k = 1,
+                           sigma = 1, s = 0.2, delta = 2.5)
> stat_simple
```

```
$Stationary_distribution
 In-control False-alarm         OOC  True-alarm
 0.68001029  0.12823186  0.01281081  0.17894704
```

This example highlights that—even though there are several generalisations implemented—it is possible to use a simple model which requires the estimation of a relatively small number of parameters. The print method omits the transition matrix and other, ancillary results, but they are nonetheless stored in the resulting object.

There are many possible setups that can be modelled using the Markovstat function. Showing an example for each would be nigh impossible and redundant. Above, the simplest Markov-chain based model was shown, next we shall use all generalisations and customisation options at once i.e., the most argument-heavy setup: the model below has exponential-geometric mixture distributed random shift size, random repair, state-and-interval-length-dependent random sampling time:

```
> stat_expgeo <- Markovstat(shiftfun = 'exp-geo', h = 1.5,
+                 k = 2, sigma = 1, s = 0.2, delta = 1.2,
+                 probmix = 0.7, probnbin = 0.8, disj = 2,
+                 RanRep = TRUE, alpha = 1, beta = 3,
+                 RanSam = TRUE, StateDep = TRUE, a = 1,
+                 b = 15, Vd = 100, V = 8)
> head(stat_expgeo[[1]])
```

```
  In-control  False-alarm   OOC_1_0.04   OOC_2_0.121
0.0000000000 0.0000000000 0.0008830967 0.0011367170
 OOC_3_0.202  OOC_4_0.283
0.0013968075 0.0016466962
```

It can be seen that the stationary distribution is expanded due to the random shift size, thus additional information is shown in the probability value labels. There is one IC and one FA state but there are multiple OOC and TA states (for all discretised distances). For the latter two the label contains the following information separated by the '_' character: state type, state index number, distance from μ_0 represented by the state (the midpoint of thee interval).

3.3 Function Markovchart

The Markovchart function is the main function of the package, used for cost calculation and optimisation. The function comes with many arguments, which is due to the fact that the function incorporates the above-described mathematical methods in a highly customisable way. This design was motivated by two main reasons: one is that even though the methods have differences in assumptions, their outputs in the form of cost-related statistics are similar. The second reason is that it allows the user to quickly test different models with the change of argument values instead of whole functions. Also, if a complex model with all generalisations is not feasible or necessary, the user can leave out generalisations and/or leave several arguments blank and the model will still work for the given options and data. The list of the most important arguments can be seen below:

- `statdist`: The stationary distribution of the Markov chain. Must be an object of class `Markov_stationary`, preferably created by `Markovstat`.
- `h`: The time between samplings. Must be a positive value, can be a numeric vector. For optimisation, this is the initial value. Inherited from `statdist` if not given.
- `OPTIM`: Logical. Should the resulting G-value (weighted average of the expected cost and cost standard deviation) be optimised by finding the adequate value of `h` and `k`.
- `k`: The control limit (critical value). Must be a positive value, can be a numeric vector. For optimisation, this is the initial value. Only one sided shifts are allowed, thus there is only one control limit. Inherited from `statdist` if not given.
- `p`: The weight of the cost expectation in the calculation of the G-value; should be between 0 and 1.
- `constantr`: Logical. Should the repair cost be assumed to constantly occur over time (`TRUE`) or assumed to only occur when there is a repair due to an alarm (`FALSE`, default)? If `TRUE`, then the repair cost should be given per unit time.
- `ooc_rep`: Numeric value between 0 and 1. The proportion of repair cost occurring during out-of-control operation. Default is 0. If a value greater than 0 is set, then `constantr` should be `TRUE`, but it is not forced.
- `cs`: Sampling cost per sampling.
- `cofun`: A function describing the relationship between the distance from the target value and the resulting out-of-control costs.
- `coparams`: Numeric vector. Parameters of `cofun`.
- `crfun`: A function describing the relationship between the distance from the target value and the resulting repair costs.
- `crparams`: Numeric vector. Parameters of `crfun`.
- `cf`: Numeric. The false alarm cost. Only relevant when `shiftfun` is `'deg'`.
- `vcrfun`: A function describing the relationship between the distance from the target value and the resulting repair cost variance.
- `vcrparams`: Numeric vector. Parameters of `vcrfun`.

The details of the cost calculation method differs between shift size distributions. Closed-form calculation of the expected squared distances between shifts is straightforward in the fixed shift size case, but have been solved for exponential shifts and exponential-geometric mixture distributions.

The returned value depends on the parameters:

If `h` and `k` are both of length 1, then the value of the function is a `Markov_chart` object, which is a list of length 4, detailing the properties of the control chart setup. Optimisation does not change the structure of the output, as only the optimal `h` and `k` parameters are evaluated in the end.

- `Results`: A named numeric vector consisting of:
  - `G-value`: The G-value as defined by Eq. 7
  - `Expected cost (C)`: $\mathrm{E}(C)$ as defined by Eq. 3
  - `Total cost std. dev.`: $\mathrm{Var}_{tot}(C)$ as defined in Eq. 6, only relevant and calculated for non-degenerate shift size distributions
  - `Cost std. dev. due to process var.`: $\mathrm{Var}(C)$
  - `Second process moment`: $\mathrm{E}(C^2)$

  - Third process moment: $\mathrm{E}(C^3)$
  - Fourth process moment: $\mathrm{E}(C^4)$
- Subcosts: Vector of sub-costs that are parts of the total expected cost. This can be especially useful when the goal is to find which cost component inflates the total cost.
- Parameters: A vector that contains the time between samplings (h) and critical value (k) which was used in the control chart setup.
- Stationary_distribution: The stationary distribution of the Markov chain. Further information about the stationary distribution can be calculated using the Markovstat function.

If either h or k have length greater than 1, then the *G*-value (weighted average of average cost and cost standard deviation, as defined at (7)) is calculated for all given values without optimisation. The value of the function in this case is a Markov_grid data frame with length(h)*length(k) number of rows and three columns for h, k and the *G*-value.

Let us take a look at the simplest case again, now with cost calculation:

```
> res_simple <- Markovchart(statdist = stat_simple, cs = 1,
+ crparams = 20, coparams = 50)
> res_simple

$Results
   G-value Expected cost Cost sd due to process var.
1 12.40682      12.40682                    18.14134
  2nd process moment 3rd process moment 4th process moment
1           483.0373           21268.82           972450.5

$Subcosts
  In-control cost False-alarm cost Out-of-control cost
1               1         2.564637           0.6405403
  True-alarm cost
1         8.20164

$Parameters
  Time between samplings (h) Critical value (k)
1                          1                  1
```

coparams and crparams are simple numeric values here, but can be numeric vectors for the more complex models.

Setting OPTIM = TRUE prompts the function to optimise with respect to the *G*-value. (As the default value of p is 1, the *G*-value here is simply the expected total cost.) Optimisation is conducted using the optimParallel package (Gerber 2020), which provides parallel thus fast computation for the L-BFGS-B method. Parallel settings are automatically set, but can be changed manually using the parallel_opt argument. In case of optimisation the given h and k parameters are only starting values, the optimal parameters can be seen in the Parameters section of the output list. The structure of the output is otherwise the same as above:

```
> res_simp_optim <- Markovchart(statdist = stat_simple,
+ OPTIM = TRUE, cs = 1,
+ crparams = 20, coparams = 50)
> res_simp_optim

$Results
   G-value Expected cost Cost sd due to process var.
1 10.73634      10.73634                    21.39669
  2nd process moment 3rd process moment 4th process moment
1           573.0874           37380.25            2502524

$Subcosts
  In-control cost False-alarm cost Out-of-control cost
1        2.166483         1.014868            1.701831
  True-alarm cost
1        5.853159

$Parameters
  Time between samplings (h) Critical value (k)
1                  0.4615776           1.933419
```

Let us also compute costs for the exponential-geometric model seen above. We will further detail our model by using a custom repair variance function and *G*-value calculation which involves the cost standard deviation with a 0.1 weight (`p = 0.9`):

```
> vcrfun_new <- function(mudist, vcrparams){
+ mudist <- mudist
+ vcrb   <- vcrparams[1]
+ vcrs   <- vcrparams[2]
+ vcrs2  <- vcrparams[3]
+ vcr    <- vcrb + vcrs / (mudist + vcrs2)
+ return(vcr)}
>
> res_expgeo <- Markovchart(statdist = stat_expgeo, cs = 1,
+                         p = 0.9, coparams = c(10, 6),
+                         crparams = c(20, 3),
+                         vcoparams = c(10000, 100),
+                         vcrfun = vcrfun_new,
+                         vcrparams = c(50000, -600000, 1.5))
> res_expgeo

$Results
   G-value Expected cost Total cost sd
1 38.04535      30.88199      102.5155
  Cost sd due to process var. 2nd process moment
1                    16.12781           1213.804
  3rd process moment 4th process moment
1           59777.51            3552291

$Subcosts
  Sampling cost Repair cost OOC cost
```

```
1     0.6666667    5.386237 24.95974

$Parameters
  Time between samplings (h) Critical value (k)
1                        1.5                  2
```

It can be seen that now the G-value and the expected cost have different values. Also, the total cost standard deviation (as defined in Eq. 6) is added to the output.

As mentioned above, the `Markovchart` function also provides a vectorised method for calculating G-values for a vector of h or k values, in which case the output is a data frame. Vectorised calculation uses parallelisation with the `doParallel` (Wallig et al. 2020b) and `foreach` (Wallig et al. 2020a) packages. In this case, the output of the function is an object of class `Markov_grid` and `data.frame` with three columns:

```
> stat_exp <- Markovstat(shiftfun = 'exp', h=0.14, k=0.3,
+                        sigma = 1, s = 0.2, delta = 2,
+                        RanRep = TRUE, alpha = 1,
+                        beta = 3, V=18)
>
> hvec     <- seq(0.14, 1, (1 - 0.14) / 15)
> kvec     <- seq(0.3,  2, (2 - 0.3)  / 15)
> Gmtx     <- Markovchart(statdist = stat_exp, h = hvec,
+                         k = kvec, p = 0.9, cs = 1,
+                         coparams = c(10, 3),
+                         crparams = c(1, 2))
> head(Gmtx,2)

     h         k    value
1 0.14 0.3000000 21.05456
2 0.14 0.4133333 20.74024
```

3.4 Method `plot.Markov_grid`

`plot.Markov_grid` is a convenience method for plotting G-values in a contour plot as the function of h and k. The method uses the `ggplot2` (Wickham et al. 2020) and the `metR` package (Campitelli 2020) for visualisation. The most important function arguments are listed below:

- `x`: A `data.frame` with three columns (preferably created by the `Markovchart` function): h, k and the G-values.
- `y`: The name of the scale.
- `xlab`: A title for the x axis.
- `ylab`: A title for the x axis.
- `low`: Colour for the low end of the gradient.
- `mid`: Colour for the midpoint.
- `high`: Colour for the high end of the gradient.
- `nbreaks`: Number of contour breaks.

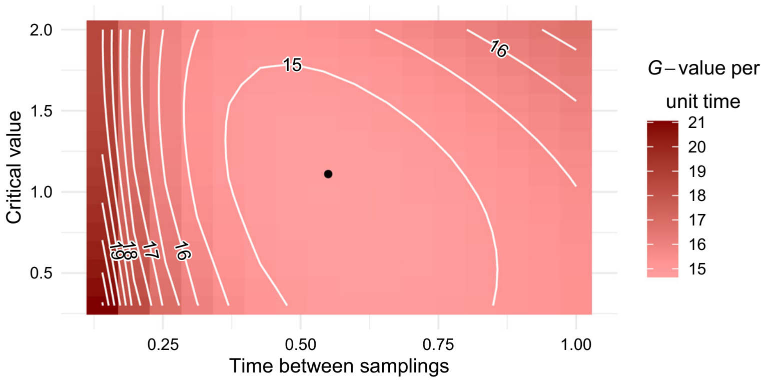

Fig. 3 Contour plot of G-values, black dot: optimal h and k parameters

The output of the function is a plot object of class `gg` and `ggplot` produced using the `ggplot2` package.

As the `Markovchart` function provides vectorised method for calculating G-values, one can easily create the data required by the `plot.Markov_grid` method. We have already assembled a dataset in the end of Sect. 3.3 and we can run the `plot.Markov_grid` plot method on it. See the example in Fig. 3.

`plot.Markov_grid` is mainly a convenience method for quick plotting and exploration of the relationship between the parameters and the resulting G-value. It can be modified like any other plot produced by the `ggplot2` package as can be seen in the added dot in the example. The data can also be modified if, for example, one would like to use a different currency.

3.5 Function `Markovsim`

The `Markovsim` function is able to simulate processes with assumptions described in Sect. 2. The same type of processes are handled by the `Markovchart` function, but in that case all calculations are analytic. The only new parameters compared to the `Markovchart` function are `num`, which is the length of the simulation measured in sampling intervals, `detail`, which is the number of simulated data points within a sampling interval (including the sampling itself) and `burnin`, which is the number of samplings deemed as a burn-in period. The burn-in period is necessary because the starting distribution is a degenerate distribution concentrated on the IC state, which is to say that the process is assumed to always start from the target value.

The output of the function is a `Markov_sim` object, which is a list of length 4:

– `Value_at_samplings`: The process value at sampling.
– `Sampling_event`: The event at sampling. The event at sampling can be success (there was a sampling but no alarm), alarm (sampling with alarm) and failure (no sampling occurred).
– `Simulation_data`: The simulated data (distances from the target value).

– Stationary_distribution: The stationary distribution of the Markov chain, created by discretising the simulated data.

Note that many, but not all features can be found here that we saw at the documentation of Markovchart. The reason behind this is that the main focus of the simulator is to be able to give support for setting up complicated models with Markovchart. In practice it was found that comparing simulations with theoretical results can be useful for parameter tuning (e.g., finding discretisation parameters that give accurate results but keep running times low). Thus simulations for trivial cases—i.e., the traditional fixed shift size model—were not implemented as there is no discretisation, random repair and random sampling which would introduce bias or uncertainty.

An example can be seen below:

```
> res_sim <- Markovsim(shiftfun = 'exp-geo', num = 10000,
+                     h = 1.5, k = 2, sigma = 1, s = 0.2,
+                     delta = 1.2, probmix = 0.7,
+                     probnbin = 0.8, disj = 2,
+                     RanRep = TRUE, alpha = 1, beta = 3,
+                     RanSam = TRUE, StateDep = TRUE, a = 1,
+                     b = 15, Vd = 100, V = 8, burnin = 500)
>
> str(res_sim, width = 60, strict.width = 'cut')
```

```
List of 4
 $ Value_at_samplings     : num [1:10000] 0 0 0 0 0 ...
 $ Sampling_event         : chr [1:10000] "failure" "fail"..
 $ Simulation_data        : num [1:1500000] 0 0 0 0 0 0 0 ..
 $ Stationary_distribution: Named num [1:200] 0 0 0.0291 0..
  ..- attr(*, "names")= chr [1:200] "In-control" "False-a"..
 - attr(*, "class")= chr [1:2] "Markov_sim" "list"
```

The first element is a numeric vector of the true process value (the one without error). The second is the event at the time of the sampling. The third element is the raw data, where not only the values at the times of the samplings but between samplings are available. (The raw data is actually the true process value without measurement error.) The fourth element is the stationary distribution of the Markov chain, created by discretising the simulated data. The stationary distribution is calculated by omitting the data from the burn-in period.

As mentioned above, the simulator can be useful to check the results of the Markovchart function. Below we will compare the previously shown exponential-geometric model with simulated results using the same parameters. A good basis of comparison is the stationary distribution, as all further results can be calculated from it. For better visualisation the stationary distributions produced by the Markovstat and Markovsim functions can be transformed in a way that does not take into account the type of the state, only the distance:

```
> Vd = 100
> distance_dist <- stat_expgeo[[1]][c(1, (Vd+2):(Vd*2))] +
+                  stat_expgeo[[1]][2:(Vd+1)]
>
```

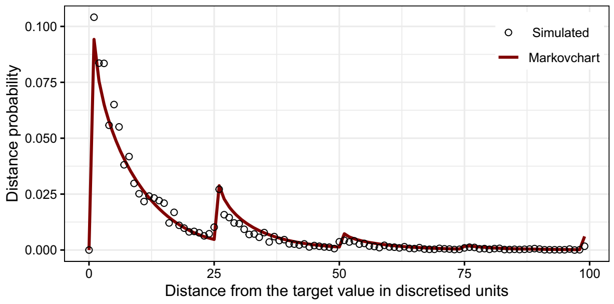

Fig. 4 Simulation and theoretical calculation comparison for the stationary (distance) distributions, simulation was run for 10,000 sampling intervals with the first 500 regarded as burn-in

```
> discr_sim      <- res_sim[[4]][c(1, (Vd+2):(Vd*2))] +
+                      res_sim[[4]][2:(Vd+1)]
```

The simplified theoretical and simulated stationary distributions can be seen in Fig. 4. The related code chunks are not implemented in the package as it is an ad-hoc comparison. Users may need a completely different measure such as costs or the fraction of alarm states.

The probability of the IC state is 0 here, due to the imperfect repair and the highest probability can be seen at the state just above the target value. The increase in probability at the furthest distance taken into account is due to the finite number of states, (since the support of the distributions in reality is not finite). The effect of the geometric distribution can clearly be seen, as the distribution is multimodal. The peaks correspond to intervals around the multiples of the geometric shift size which is 2 (`probmix = 0.7, disj = 2`). Following the peaks, a near-exponential decrease can be seen in the probability. This can be attributed to the effect of exponentially distributed shifts added to the discrete shift. The shape of the repair size distribution in this setup is also such, that it promotes repairs relatively close to the target value (`alpha = 1, beta = 3`). We can assess that the simulated and theoretical results fit each other well. It can be noted that the fit depends on the discretisation parameters used, i.e., `V` and `V_d` should be sufficiently large to reach good results. What constitutes as sufficient changes from problem to problem: Different parameters should be evaluated and the resulting stationary distribution inspected. A good rule of thumb is that the sum of the remaining probabilities (above `V`) should be relatively low. In other words, the bump at the last probability should be small on the figure. Regarding the value of `V_d`, a minimum of 30 states is advisable in most cases.

4 Model comparisons

It was mentioned previously that the `edcc` `R` package provides some features that can also be found in the `Markovchart` package. The authors of the `edcc` package have successfully validated their method in Zhu and Park (2013a), thus it provides a good basis for comparison. The intersection of the two packages are models that use the assumptions of Duncan's traditional cycle model with a sample size of one in a one-sided shift case. In the Markovchart package it corresponds the use of fixed shift size (shiftfun = 'deg'). Let us now compare the results of the two package based on the fixed shift size model shown in Sect. 3.3:

```
> res_edcc <- ecoXbar(C0 = 0, C1 = 50, Cf = 20, b = 1, n = 1,
+                  delta = 2.5, lambda = 0.2, d1 = 0, d2 = 0,
+                  T0 = 0, Tc = 0, Tf = 0, Tr = 0, Cr = 20,
+                  a = 0, sided = 'one', par = c(1, 1))
>
> res_edcc

$optimum
 Optimum h  Optimum L  Optimum n        ECH
 0.4616145  1.9332665  1.0000000 10.7363411

$cost.frame
 Optimum h Optimum L Optimum n      ECH
 0.4616145  1.933266         1 10.73634

$FAR
[1] 0.05500826

$ATS
[1] 0.4187628

> res_simp_optim

$Results
   G-value Expected cost Cost sd due to process var.
1 10.73634      10.73634                    21.39669
  2nd process moment 3rd process moment 4th process moment
1           573.0874           37380.25            2502524

$Subcosts
  In-control cost False-alarm cost Out-of-control cost
1        2.166483         1.014868            1.701831
  True-alarm cost
1        5.853159

$Parameters
  Time between samplings (h) Critical value (k)
1                  0.4615776           1.933419
```

Both models optimise the free parameters with respect to the total expected costs. Thus it is straightforward to compare the values of ECH vs. Expected cost (C), Optimum h vs. Time between samplings (h) and Optimum L vs. Critical value (k) in the edcc and the Markovchart package results respectively. Based on these, the packages provide near-identical results which further proves that the packages provide two different approaches to the same underlying process.

As the Markov chain-based model has many parameters, description of their relationship is a high dimensional problem. Many associations between parameters and the resulting expected costs and cost standard deviations have already been described in Zempléni et al. (2004), Dobi and Zempléni (2019a, b). Reproducing all of the previous results would be redundant and out of the scope of this article, thus here we only give a brief summary of the most important findings and highlight some new ones.

The most important difference between models is the shift size distribution as it governs the deterioration. The fixed shift size distribution represents its own category and cannot be compared directly to the exponential and the mixture distribution. Zempléni et al. (2004) analysed this classical case in detail and their results agree with traditional control chart theory. They found that the optimal time between samplings is less for higher OOC costs and optimal critical values increase with shift size. They also found that lower shift frequency leads to longer optimal sampling intervals. The expected costs of the optimal parametrisation showed a decreasing trend for increasing shift sizes. This trend can be explained by the increasing difficulty in detecting smaller shifts, which may lead to more incorrect decisions.

Dobi and Zempléni (2019b) compared exponential and exponential-geometric shift size distributions. The inspection of the stationary distributions revealed markedly different shapes, where the effect of the shift size distribution is clearly visible, as we could also see here, on Fig. 4. Their results also show that practically only the first two moments are important—although it could be suspected that higher moments would play a role as well. The relationship between the optimal parameters and the resulting expected costs and cost standard deviations were similar for both distributions.

The relationship between the parameters and the results in the recently-developed models were analysed by Dobi and Zempléni (2019a, b). In cost-optimised models they found that typically the critical value increases and the time between samplings decreases with the increased expected shift size. This result is the opposite of what was seen with fixed shift sizes. The difference can be explained by the fact that random shift sizes allow stacking of shifts, while the fixed shift size does not. The stacking of great shifts can easily overshadow the benefit of easier detection. Other parameters may play a role as well (e.g., repair costs). They also found that higher expected shift sizes entailed higher expected costs and cost standard deviations. It was seen that with the increase of the OOC cost the critical value usually stagnated, the time between samplings decreased, and the expected cost and the cost standard deviation increased. The incorporation of the cost standard deviation in the optimisation procedure substantially lowered the standard deviation while the cost expectation barely increased. Namely costs standard deviations could almost be halved in the tested scenarios while the expected cost only increased by roughly 20%. This is important, because it shows that the cost standard deviation can be lowered sometimes quite substantially while the expected cost is only moderately increased.

An important part of health care applications of the control chart model is the choice of therapy. Health care professionals can often choose from a range of therapies which have different effectiveness and entail different costs. In the model this is essentially a set of additional free parameters, namely the repair size distribution parameters, the repair cost function and its parameters and any other aspect of the process that may be connected to the therapy. Separate models can be defined for different therapies and compared: e.g., a cheap therapy may not be cost-optimal in the long run if it is ineffective and thus generates huge costs due to a badly managed chronic illness. We discuss this topic in detail in Sect. 5 below.

5 Application example

In this section we will apply the `Markovchart` package to randomised real-world data of diabetic patients.

5.1 Patient data

Our analysis is based on a month-aggregated time series data of diabetic patients from Hungary which was gathered from the period of 2007 September–2017 September. The data came from two sources: the National Health Insurance Fund of Hungary and the South-Pest Central Hospital. The first source provided information about diagnoses, treatments, health care event and related costs while the latter provided laboratory data regarding blood sugar level. Patients with International Classification of Diseases (ICD) codes (diagnosis) of E10, E11 and E14, and at least one blood sugar measurement were included initially. Only the data of patients with at least one E11 (type II diabetes) diagnosis in the study period was kept. An additional homogenising filter was the requirement of age above 40 at the time of the first diagnosis. Disease progression and therapy effectiveness estimation required at least two blood sugar (HbA1c) measurements with simultaneous therapy data. A total of 4434 patients satisfied all conditions out of which 2151 had at least two HbA1c measurements.

The example study focused on two therapy types: insulin analogues (artificial insulins) and glucagon-like peptides (GLP, promotes insulin secretion). Of course there are more treatment types, the database also lists oral antidiabetics (OAD) and human insulins, but we choose to make the analysis simpler by focusing on GLP and analogue therapies. For the sake of comparison the therapies are grouped in this way: the first group is insulin analogues with possible parallel OAD therapies but human insulin and GLP excluded. The second group is GLP therapies with possible parallel OAD and insulin analogue therapies but human insulin excluded. Essentially we are comparing the effect and cost of insulin analogues with the effect and cost of additional GLP therapies.

The monitored characteristic of the control chart is the blood sugar level, namely the HbA1c level. This is the glycated haemoglobin measured in percent and is slow-changing. The medical aspects of the disease, therapies and blood sugar measurement will not be discussed further as these fall outside of the scope of this example.

The data contains sensitive, patient level information, thus we would only able to show figures, statistics and aggregated data in the paper. However, we pseudonymised and randomised the data in a way that creates fake data but still retains most of the important characteristics and the connections between variables. Namely, random

Table 1 Number of patients in the randomised dataset

Patient group	Total	Analogue	GLP
Total (all have E11 & are over 40)	800	630	170
E11 ICD & over 40 & at least two HbA1c	492	272	99

numbers were added (or subtracted) to the date of sampling, the number of sampling per month, costs, HbA1c measurements (both average and standard deviation) and the age of the patient. Furthermore, a subsample was taken from the available patient sample to not even keep the order of patients. The otherwise often very complicated therapeutic regimens were simplified into GLP and analogue categories. This simplification sometimes meant the overwriting of the true therapy used. This is meant to further complicate the identification of patient by e.g., therapeutic pattern. The number of sample elements in the final, randomized dataset can be seen in Table 1.

5.2 Parameter estimation

This section will provide succinct information about parameter estimation and thus the disease, therapies and the related costs. The aim of the application is to show as many aspects of the `Markovchart` package to an `R` user as possible. Due to this, some estimates come from simplified calculations. During industrial or academic applications a group of data scientists and healthcare professionals could provide more accurate estimates, but the work of such a research group is out of the scope of this paper. For cost calculations, the 2021 March 21 EUR-HUF exchange rate was used (1 EUR = 369.05 HUF).

Parameters related to disease progression, namely the expected number of shifts per unit time (days) (inverse of shift intensity argument `s` in the `Markovchart` function) and the expected shift size δ (`delta` argument in the `Markovchart` function) were estimated using the time series data of HbA1c measurements. The δ estimate was calculated from a filtered dataset where we required a HbA1c change larger than 2σ and more than 90 but less than 184 days between samplings. The former was necessary to find actual shifts and not just random fluctuations and the latter to try to estimate the size of one shift and not the size of multiple stacked shifts. The expected shift size was $\delta = 1.16\%$. The restriction during estimation of the expected time between samplings was less strict: it was required that the samplings should be at maximum one year ($<$367 days to account for leap years) from each other. The time between shifts were then gathered from this filtered database and averaged. The shift intensity was calculated by taking the reciprocal of this average and was $s = 0.0045$.

Measurement error (the process standard deviation σ, `sigma` argument in the `Markovchart` function) was estimated using lucky anomalies: the HbA1c level should not be measured more frequently than three months, because the measured values should barely change, and, in relation to this, only 4 measurements are supported per year by the National Health Insurance Fund of Hungary. Nonetheless there were 221 cases in the original data set where HbA1c measurement occurred more than

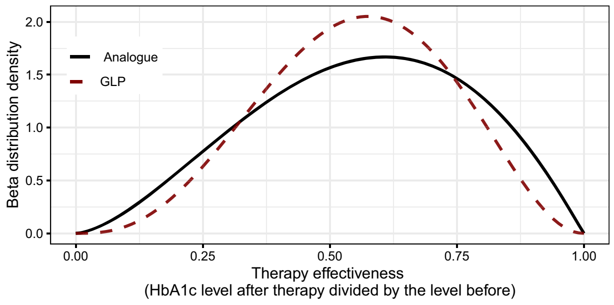

Fig. 5 Therapy effectiveness comparison using the estimated beta distributions

once per month. This number was boosted in the randomised dataset to 692. This data essentially provides information about the measurement error as the actual HbA1c level should change only mildly within such a short time frame. Our estimate with this simple method was $\sigma = 0.34$. It is difficult to compare this result with existing literature due to different methodologies but our estimate is close to the one calculated by Phillipov and Phillips (2001) (even on the randomised data).

Therapy effectiveness was estimated using the definition of effectiveness in the Markov chain-based control charts: the proportion of distance from the target value after treatment compared to the distance before. The target value was set to be 4% HbA1c level, which is the lowest healthy level (Wang 2017). Setting it to a higher value would exclude data, as the target value is the lowest considered. When estimating therapy effectiveness, after the initial filtering of the data (see Sect. 5.1), we also restricted HbA1c data to cases where the initial value (after which improvement was seen) was 6%. This was because we wanted to see effect of the therapies only in cases where there is some notable deviation from the healthy state. Improvement was defined as $> 2\sigma$ (`sigma` argument in the `Markovchart` function). Again, values lower than this threshold were considered stagnation (which could also be caused by an effective therapy followed by a degradation, thus making it unreliable) and were not included in the effectiveness estimation. To minimise autocorrelation and the effect of degradation parallel to therapies, only samplings where where more than 90, but less than 184 days elapsed between the two were considered. Parameters of the repair size beta distributions were estimated from the mean and the variance of the ratios of consecutive HbA1c values between samplings. The estimated distribution for analogue therapies was $beta(2.63, 2.05)$ and for the GLP therapies $beta(3.85, 3.11)$. The estimated beta distributions for both therapies can be seen in Fig. 5.

It can be seen that the GLP therapy is marginally better based on this dataset than the analogue therapy (judging by the mode). The effectiveness of analogue therapies have a somewhat greater variance. Of course, this method does not take into account the HbA1c level itself, only the change (as a proportion).

Random sampling (i.e., patient non-compliance) was not taken into account as our data did not provide information about this. Nonetheless a sensitivity analysis could be conducted to see the effect of different hypothesised compliance levels, but we choose not to complicate the scenario with such an analysis.

For daily therapy cost estimation (on top of the initial filtering) we also restricted the data to cases where HbA1c levels were less than or equal to 10%. We observed that cost data as a function of HbA1c level becomes unreliable above this value. This may be due to several factors, one being that patients in highly deteriorated states might have other illnesses (comorbidities) and complications. HbA1c values in between-sampling time periods were imputed using linear interpolation. This was necessary, as therapies were ongoing even between samplings. To be able to estimate variances, the HbA1c level was discretised into 150 categories. The relationship between the therapy costs, costs variances and the HbA1c level was then estimated using (non-)linear least squares. This was accomplished with the `R` function `nls`. GLP therapy and complication (i.e., OOC) costs in relation of the HbA1c level were estimated and calculated using the default cost and cost variance functions, discussed in Sect. 2.4. However, the analogue therapy cost and cost variance estimation used a function of the form

$$c_r(v) = c_{rb} + \frac{c_{rs_1}}{v + c_{rs_2}},$$

where v is again the distance from the target value. The resulting functions were the following (everything is given in euro):

$$c_{analogue}(v) = 3.3 + \frac{-12.22}{v + 9.37}, \; c_{GLP}(v) = 1.69 + 0.07v,$$
$$\varsigma_{analogue}(v) = 7.09 + \frac{-3.32}{v + 1.08}, \; \varsigma_{GLP}(v) = 7 - 0.2v,$$

where v is the HbA1c level.

The same restrictions, discretisation and `nls` function was used when estimating the relationship between the OOC (health care event) cost, cost variance and HbA1c level. Additionally, as there may be a lag between a deteriorated patient state and the resulting health care event, a 6-month cumulative cost was calculated. The resulting functions were the following:

$$c_o(v) = 0.62 + 0.0062A(v)_h^2, \; \varsigma_o(v) = 5.12 + 0.13A(v)_h^2,$$

where v again is the HbA1c level.

The resulting lines fitted to the data together with standard deviation bands can be seen in Fig. 6.

It can be seen that analogue and GLP therapies have similar costs, while compared to these, the OOC costs (especially for higher HbA1c levels) are considerably lower. This is expected as diabetes requires constant treatment while complications may or may not occur. Also, it is understandable to keep patients relatively healthy (thus complication-free) even at a high treatment cost.

There are some more parameters that are worth mentioning: for model fit time between samplings and the critical value was also estimated from the data. h was the

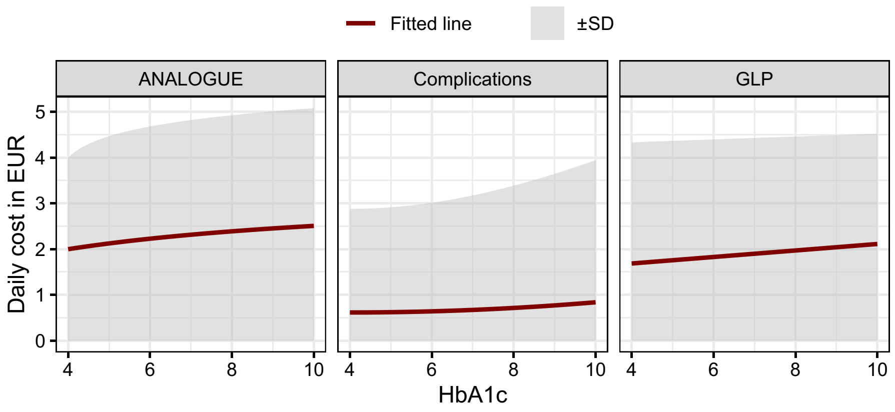

Fig. 6 Therapy and medical complication costs (€)

mean time between samplings (206.22 days), calculated from the randomised patient data. k was determined according to the medical guidelines (7%, see Ton et al. 2013). `constantr = TRUE` and `ooc_rep = 1` (p_r in Eq. 5) were used, as the therapy is constant and occurs even if there is no alarm (alarm states still play an important role as HbA1c reduction can only be initiated by an alarm).

5.3 Stationary distribution and cost estimation with the Markovchart function

Before we can feed the estimated parameters to the `Markovchart` function, we have to define the above-mentioned custom cost and cost variance functions in R. Afterwards we can run the `Markovchart` function for both the analogue and the GLP therapy group.

```
> crfun_ANALOGUE <- function(mudist, crparams){
+   mudist <- mudist
+   crb    <- crparams[1]
+   crs    <- crparams[2]
+   crs2   <- crparams[3]
+   cr     <- crb + crs / (mudist + crs2)
+   return(cr)}
> vcrfun_ANALOGUE <- function(mudist, vcrparams){
+   mudist <- mudist
+   vcrb   <- vcrparams[1]
+   vcrs   <- vcrparams[2]
+   vcrs2  <- vcrparams[3]
+   vcr    <- vcrb + vcrs / (mudist + vcrs2)
+   return(vcr)}
>
> stat_ANALOGUE <- Markovstat(
+   shiftfun = 'exp', h = 206.22, k = 3,
+   sigma = sigma_param, s = s_param,
+   delta = delta_param, RanRep = TRUE,
+   alpha = as.numeric(ANALOGUE[1]),
+   beta = as.numeric(ANALOGUE[2]),
+   Vd = 100, V = 18)
> res_ANALOGUE <- Markovchart(
+   statdist = stat_ANALOGUE, p = 1,
+   constantr = TRUE, ooc_rep = 1,
+   cs = sampling_cost,
+   coparams = summary(mod.COST)$coef[ , 1],
+   crfun = crfun_ANALOGUE,
+   crparams = summary(mod.ANALOGUE)$coef[ , 1],
+   vcoparams = summary(mod_var.COST)$coef[ , 1],
+   vcrfun = vcrfun_ANALOGUE,
+   vcrparams = summary(mod_var.ANALOGUE)$coef[ , 1])
>
> stat_GLP <- Markovstat(
+   shiftfun = 'exp', h = 206.22, k = 3,
+   sigma = sigma_param, s = s_param,
+   delta = delta_param, RanRep = TRUE,
+   alpha = as.numeric(GLP[1]),
+   beta = as.numeric(GLP[2]),
+   Vd = 100, V = 18)
> res_GLP <- Markovchart(
+   statdist = stat_GLP, p = 1,
+   constantr = TRUE, ooc_rep = 1,
+   cs = sampling_cost,
+   coparams = summary(mod.COST)$coef[ , 1],
+   crparams = summary(mod.GLP)$coef[ , 1],
+   vcoparams = summary(mod_var.COST)$coef[ , 1],
+   vcrparams = summary(mod_var.GLP)$coef[ , 1])
```

Before inspecting the cost elements of the results, it is useful to check the stationary distribution of the Markov chains and compare them to the empirical HbA1c data. We can do this in a visually appealing fashion by creating a distance distribution from the stationary distribution (by not taking into account the type of the state, only the distance, as we did in Sect. 3.5). The comparison can be seen in Fig. 7.

We can assess that the stationary distribution fits the data at an acceptable level. This is actually a very beneficial result, as the main focus of the function is cost estimation which is based on the stationary distribution.

Now that we have assured that there are no anomalies in the stationary distribution estimation, let us check the expected costs and costs variances. The costs statistics are shown in Table 2.

One can see that the total expected cost per day is 3.17€ for the analogue and 2.78€ for the GLP group, with considerable standard deviation (>3.7€) in both cases. (Note that `p = 1`, thus the G-value is simply the expected cost.) The main source

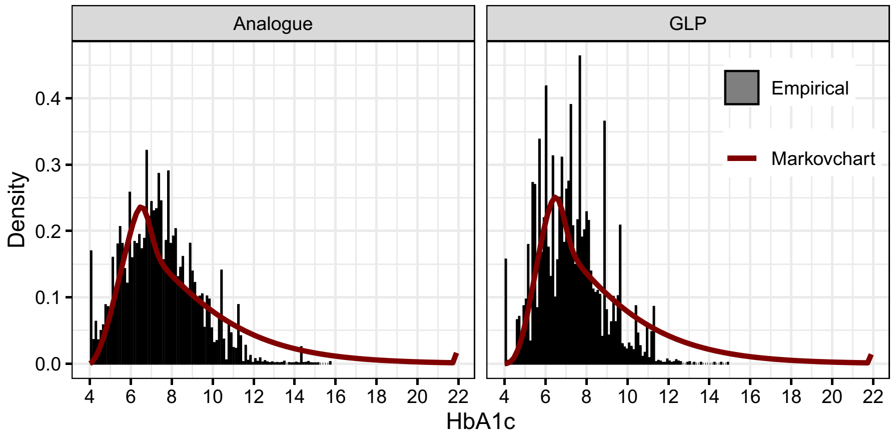

Fig. 7 Empirical and theoretical stationary distributions

Table 2 Total expected costs, sub-costs and cost standard deviation (€)

Data/therapy	Analogue			GLP		
Empirical	Average cost			Average cost		
	4.18			2.36		
	Sampling	Therapy	Event	Sampling	Therapy	Event
	0.04	2.32	0.8	0.04	1.88	0.43
	Cost standard deviation			Cost standard deviation		
	4.35			3.49		
Markovchart	Expected cost			Expected cost		
	3.17			2.78		
	Sampling	Therapy	Event	Sampling	Therapy	Event
	0.04	2.39	0.74	0.04	2	0.73
	Cost standard deviation			Cost standard deviation		
	3.75			3.71		

of the costs is the therapy cost, as we have seen during the parameter estimation. We can also compare the results to the empirical data for additional goodness of fit evaluation. It can be seen that the estimates of the `Markovchart` function are quite close to the ones estimated directly from the empirical data. Naturally there is room for improvement in some cases. For example the difference in the event costs is quite clear between therapies from the empirical data, while the `Markovchart` estimates do not reflect this. This is mainly an assumption problem: it is assumed that event (i.e., OOC) costs solely depend on the states of the chain (i.e., HbA1c level), thus due to the similar stationary distributions (see Fig. 7) there will not be much difference between the event costs either. This of course can be solved by separate event cost estimates in the patient groups but since the absolute difference is small it is not important in this case. The sampling costs are exactly the same, as the empirical mean

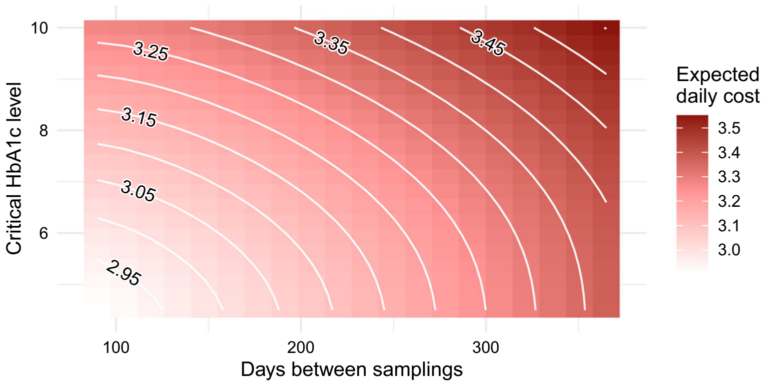

Fig. 8 Contour plot of expected costs (€) related to analogue therapy

time between samplings (estimated from the whole patient population) was used in all cases. Overall, we can say that both the stationary distributions and the costs fit the data at an acceptable level. This is useful, because it means that we have a good baseline model which allows scenario exploration (i.e., changing various parameters to check their effects on the costs).

We will now use the vectorisation feature of the `Markovchart` function and explore how the expected daily cost changes with different days between samplings and critical HbA1c levels. The code is almost the same as before, only the value of `h` and `k` is changed to numeric vectors.

We can use the `plot` method to quickly assess the results. The contour plot for the analogue therapy patient group is presented in Fig. 8.

It can be seen that more frequent samplings and lower critical values entail less daily costs on average. The optimal parameters fall outside of the plotted area. This is due to the fact that lower parameter values are not viable. Namely, HbA1c measurements within 90 days of each other become redundant due to the slow variation and HbA1c values lower than 4% are actually indicating too low blood sugar. Nonetheless the results provide useful information about the relationship between the parameters, for example we can see that the expected daily costs is less sensitive on the differences between greater parameter values.

Since we have two different therapies it may be beneficial to compare them. We could simply plot the data of the GLP patient group with the plot method but since there are only two therapies a 3D might provide better understanding of the relationships. Figure 9 contains the contour plots of both therapy groups. (The implementation of this plotting system into the package was considered but was eventually discarded as it is convenient only in rare cases and requires a lot of fiddling to achieve a visually appealing result.)

We can see the same contour plot on the upper half of the figure, as before in Fig. 8. In addition, GLM therapy related costs are shown in the lower part. It is clear that the total expected daily cost is higher in case of analogue therapy for all h and k combinations. Costs related to both therapies are the cheapest when the time between samplings and the critical HbA1c level is as low as possible. However, we can also assess that GLP and

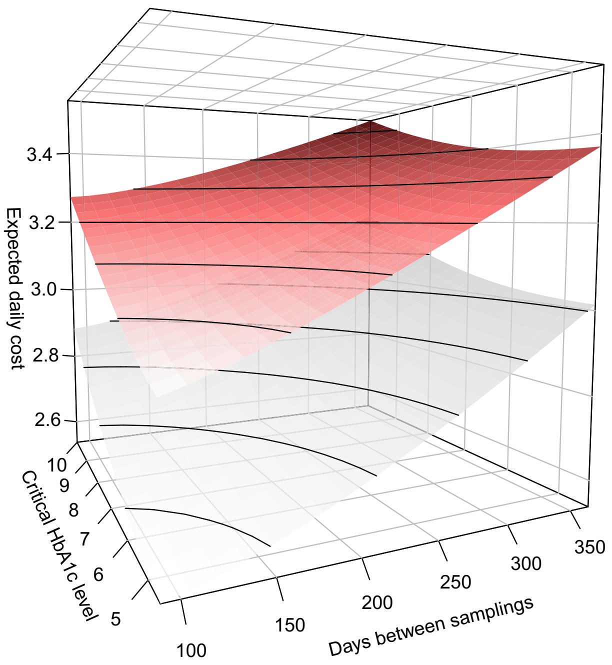

Fig. 9 Contour plot of expected costs (€) related to analogue (above) and GLP therapy (below)

analogue therapy-related costs are similarly sensitive to the parameters (see contour lines). This is a potentially important information for health care professionals and insurance companies in setting up cost-efficient therapeutic and monitoring regimens. It can be noted that this is a cost effectiveness viewpoint based on randomised data.

Cost standard deviation can play an important role in decision making, thus it is beneficial to inspect this aspect too. By changing the `p = 1` parameter in the `Markovchart` function to `p = 0`, we can create the same figure, but now the z axis will be the cost standard deviation related to therapies. The results can be seen in Fig. 10. The figure uses the same colour coding as before. The difference of the costs standard deviations between therapies is much less than what we saw with expected costs. It can also be seen that the costs standard deviation are not very sensitive on the parameters (the range of standard deviations is roughly 3.3–4.3€). The most important—and most visible—result is the overlapping surfaces (contours), which means that order of therapies in the sense of the higher cost standard deviation depends on the free parameters. In other words, when both the time between samplings and the critical HbA1c value are relatively low the analogue therapy performs better cost standard deviation wise, but at higher parameter values the GLP therapy is a better choice. This result highlight the importance of multidimensional analysis and the usability of the functions provided by the package. Again, this is a cost effectiveness viewpoint based on randomised data and may not hold for a real-life situation or data.

It is useful to conduct a sensitivity analysis on the most important parameters. The above models assume that repair cost at OOC states (thus when there is no alarm and

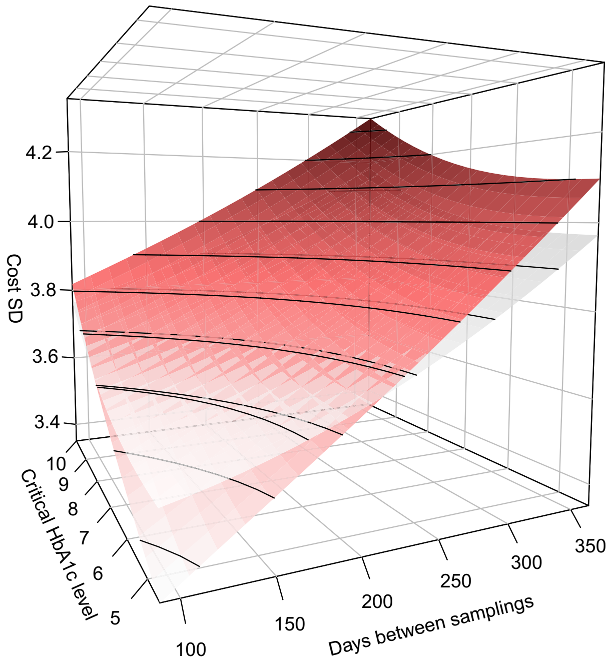

Fig. 10 Contour plot of cost standard deviations (€) related to analogue (above towards the back of the image) and GLP therapy (above towards the front of the image)

no intervention) are the same as during an alarm state (`ooc_rep = 1`). We can refine these models by assuming that intervention costs more than the usual treatment. We will focus on the insulin analogue treatment. One can use the `ooc_rep` argument to set the percentage of repair cost occurring during OOC operation. For now, we will set it to be 0.67, meaning that two-third of the repair cost will occur during OOC operation instead of 100%. The effect of sampling cost should also be inspected as the current one only takes into account strictly the cost directly related to the inspection. There could be additional hidden cost related to sampling such as equipment maintenance. We shall first assess the effect of a ten-fold sampling cost. Figure 11 shows the results with the modified parameters (note that the axes have different units than previously).

It can be seen that the optimal h and k parameters (black dot) now appear in the viable field of values. As alarm states (i.e., intervention/therapy change) costs more than the 'usual' therapy, the model discourages too low critical values, while increased sampling costs result in more time between samplings then previously seen. The positive, linear relationship between the parameters is interesting: as the time between samplings increase one should use higher critical values to minimise costs. This can be explained by the joint effect of constant (but not fixed) therapy costs and relatively low OOC (i.e., healthcare event) costs: namely, when a patient visits less often, the probability of a deteriorated state increases and this generates higher costs. To evade even higher costs due to intervention, the critical value needs to be increased. Of course this is a strictly cost-efficient viewpoint and not a medical one.

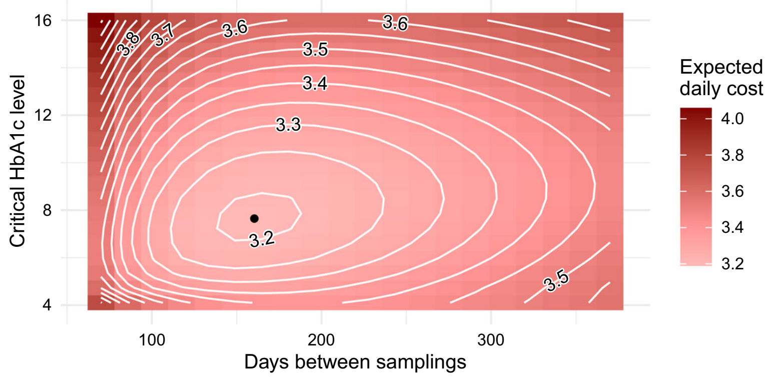

Fig. 11 Contour plot of expected costs (€) related to analogue therapy with modified `ooc_rep = 0.67` and `cs = sampling_cost * 10` parameters

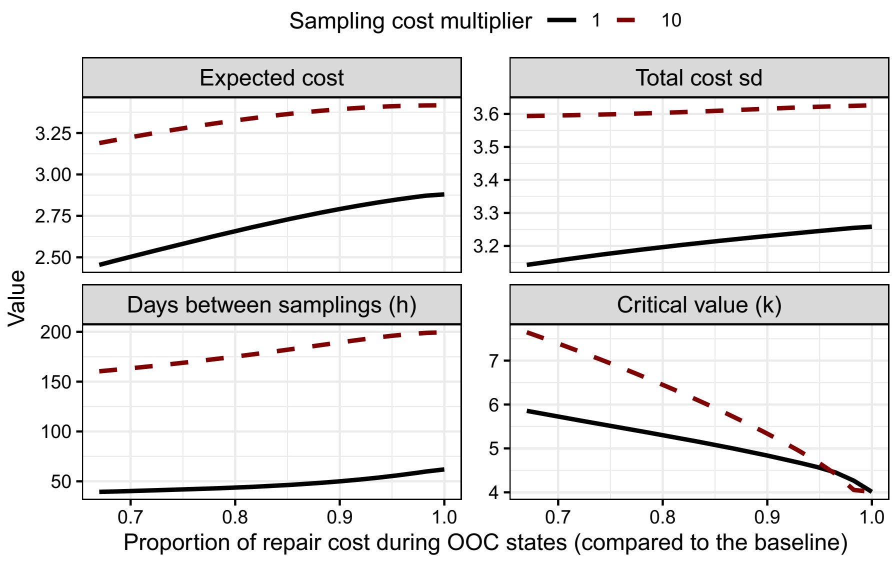

Fig. 12 Relationship between the optimal parameters and costs (€) in case of insulin analogues

How does `ooc_rep` and `cs` affect the optimal parameters and the resulting expected cost and cost standard deviation in general (between the investigated values)? We can use the `Markovchart` function on different parameter setups and visualise the results as seen in Fig. 12.

The y axis represents the percentage of the repair cost occurring during OOC operation (`ooc_rep`) on a continuous scale. The colour of the lines depends on the sampling cost (`cs`) multiplier used. The x axis is different on each plot, giving information about the value of an optimal parameter or result. The original model (as seen in Fig. 8) corresponds to the rightmost (`ooc_rep = 1`) part of the figures and solid lines. We can see that if above around 97% (`ooc_rep > 0.97`) of the repair costs also occur during OOC operation the model becomes somewhat unstable. The optimal critical value plummets to the lowest possible value. In this scenario there is no reason not to constantly try different therapies on a patient (the treatment cost will occur anyway). However, when `ooc_rep < 0.97` then the model produces non-extreme optimal parameter values. The scenario seen in Fig. 11 corresponds to the leftmost part of the plots and dashed lines. If we check the values in-between, we can see that the `ooc_rep` parameter substantially affects the optimal parameters if `cs` is increased. Namely, in this case lower `ooc_rep` values entail more days between samplings and higher critical HbA1c values. We can also see that higher sampling costs generally increase the optimal parameters and the resulting costs and cost standard deviations. The `ooc_rep` parameter only moderately affects the resulting expected cost and cost standard deviation.

6 Conclusion

As data become available in never-before-seen abundance in many fields, it is important to develop tools for their proper analysis. The `Markovchart` package aims to give health care professionals and biostatisticians a tool to model, simulate and visualise complex processes involving disease development, treatment, patient non-compliance and related costs. The package's main use is the development of cost-optimal monitoring and treatment regimens, focusing on individual patients (and homogeneous patient groups), by optimising the time between samplings and the treatment-inducing critical disease level. Even though the generalisations were developed with health care applications in mind, many other areas could benefit from the models, as the package is an `R` implementation of the Markov chain-based cost-optimal control charts developed in more-general-focus papers in Zempléni et al. (2004), Dobi and Zempléni (2019a, b).

The most important generalisations implemented in the package are the random shift size (degradation), random repair (treatment) and random sampling time (patient non-compliance). The package is thus capable of modelling and simulating many different illnesses and therapies. The simplest models are very similar to Duncan's cycle model (Duncan 1956) which were also implemented in the `edcc` package (Zhu and Park 2013b). In this simplest case the results of the two packages are virtually identical. We have shown through an application on diabetes data that the package's modelling and cost calculation tools are capable of highlighting non-trivial relationships between the therapies, time between samplings, critical value, costs and cost standard deviations.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00180-021-01175-3.

Acknowledgements The authors would like to express their gratitude towards the Healthware Consulting Ltd. (Budapest, Hungary) for their help in data preparation and providing professional knowledge during the writing of Sect. 5.

Funding Open access funding provided by Eötvös Loránd University. The work was supported by the project EFOP-3.6.2-16-2017-00015, which was supported by the European Union and co-financed by the European Social Fund.

Data availability The randomised patient data used in this study is available in the Markovchart R package and at the Harvard Dataverse Repository at https://doi.org/10.7910/DVN/UTYIOL.

Code availability The R code used during the writing of this article can be downloaded and executed using Online Resource 1 and 2.

Declarations

Conflict of interest All authors have stated that there are no conflicts of interest connected with this article.

Ethics approval All procedures performed in this study involving human participants were in accordance with the ethical standards of the Medical Research Council—National Scientific and Ethical Committee, Budapest, Hungary, and with the 1964 Helsinki declaration and its later amendments. Data were handled in accordance with personal data protection regulations. The study was approved by the National Scientific and Ethical Committee (No. IV/9693-1/2020/EKU).

A Out of control cost for different shift size distributions

The goal of the calculations below is to find the total accrued costs generated by an out-of-control process in time h, given that the starting distance from the target value is j. The cost itself is a function of the squared distance from the target value, thus the value we are looking for is actually an area under the curve from the starting time until h. For further information, see Dobi and Zempléni (2019b, 2021).

Exponentially distributed shift size Let us assume first that the shift times form a homogeneous Poisson process, and the size of a single shift is exponentially distributed, independently of previous events. The number of shifts is modelled by a Poisson distribution, with parameter ts—the expected number of shifts per unit time multiplied by the time elapsed. The shift size distribution for k shifts is a special case of the gamma distribution, the Erlang distribution $E(k, \frac{1}{\delta})$, which is the sum of k independent exponential variates each with mean δ.

Applying the formula proposed by Dobi and Zempléni (2019b) we can calculate the total squared cost incurred:

$$\begin{aligned} \mathcal{C}^2_{h,j} &= \int_0^h \left[e^{-ts} j^2 + \left(\sum_{k=1}^{\infty} \frac{(ts)^k e^{-ts}}{k!} \cdot \int_0^{\infty} (x+j)^2 \frac{(1/\delta)^k x^{k-1} e^{-x/\delta}}{(k-1)!} dx \right) \right] dt \\ &= \int_0^h e^{-ts} j^2 + \sum_{k=1}^{\infty} \frac{(ts)^k e^{-ts}}{k!} \left(k\delta^2 + (k\delta + j)^2 \right) dt \end{aligned}$$

$$= \int_0^h 2\delta^2 ts + (\delta ts + j)^2 dt = h^2 s\delta\Big(\delta + \frac{hs\delta}{3} + j\Big) + hj^2,$$

where first we have used the law of total expectation—the condition being the number of shifts within the interval. If there is no shift, then the distance is not increased between samplings, this case is included by the $e^{-ts}j^2$ term before the inner integral. Note that the inner integral is just $E(X+j)^2$ for a gamma—namely an Erlang$(k, \frac{1}{\delta})$—distributed random variable. When calculating the sum, we used the known formulas for $E(Y^2)$, $E(Y)$ and the Poisson distribution itself—where Y is a Poisson(ts) distributed random variable. $\frac{\mathcal{C}^2_{h,j}}{h}$ can be used to calculate the average cost per unit time generated by the process.

Mixture distribution as the shift size Let us assume now that the size of a single shift has a distribution which is a mixture of an exponential and a geometric distribution:

$$F_m(x) = \zeta F_g(x) + (1-\zeta)F_e(x),$$

where $\zeta \in [0,1]$ is the mixing parameter, $F_g()$ is the CDF of the geometric distribution, $F_e()$ is the CDF of the exponential distribution, and $F_m()$ denotes the CDF of the resulting mixture distribution. Such a distribution can model processes with slow degradation, mixed with sudden jumps. Different definitions are available for the geometric distribution, the one used here has support over $\{1, 2, \ldots\}$, thus $F_g(x) = 1-(1-\xi)^x$, with probability parameter ξ. A negative binomial distribution can be defined as the sum of independent geometrically distributed random variables with the same parameter. Its PMF can be written in the following way:

$$f_{nb}(x) = \binom{x-1}{x-r} \cdot \xi^r(1-\xi)^{x-r}, \tag{9}$$

where r is the number of summed geometrically distributed variables. The support of this distribution is $x \in \{r, r+1, r+2, \ldots\}$.

Let us assume now that k shifts occurred in a given time interval, and r of these were geometrically, while the rest were exponentially distributed. The previously used formula can also be used with this new shift size distribution, but the calculation of a closed form using the density function is impossible in this case, because it does not exist for this mixed distribution. Thus, some reformulation is needed. For the following paragraphs, let X denote an $E(k-r, \frac{1}{\delta})$ (Erlang) distributed random variable, which is the sum of $n-r$ independent exponential variates each with mean δ, and let Y denote a $NegBin(r, \xi)$ (negative binomial) distributed random variable, which is the sum of r independent geometric variates each with parameter ξ. Note, that for the negative binomial distribution we are using the definition given at Eq. (9). Also, for practical use, one may wish to have discrete jump points other than the integers defined by the support of the negative binomial distribution (as it is implemented in the `Markovchart` package). We can simply multiply the random variable by a constant (let us denote it by J) to create arbitrary jump-intervals. The initial equation thus takes the form

$$\mathcal{C}_{h,j}^2 = \int_0^h \left[e^{-ts} j^2 + \left(\sum_{k=1}^{\infty} \frac{(ts)^k e^{-ts}}{k!} \cdot E((X + JY + j)^2) \right) \right] dt$$

The innermost expectation, which was previously simply the expectation of $(X + j)^2$, now is replaced by $E((X + JY + j)^2)$. Observe that every other part of the equation stays the same. It is easier to present the calculation in parts, as it will result in a long expression. Firstly,

$$\begin{aligned} &E((X + JY + j)^2) \\ &= \sum_{r=0}^{k} \binom{k}{r} \zeta^r (1-\zeta)^{k-r} \Bigg[\left(\frac{k-r}{1/\delta} \right)^2 + \frac{k-r}{1/\delta^2} + \left(J \frac{r}{\xi} \right)^2 + J^2 r \frac{1-\xi}{\xi^2} + j^2 \\ &\qquad + 2 \frac{k-r}{1/\delta} J \frac{r}{\xi} + 2 \frac{k-r}{1/\delta} j + 2J \frac{r}{\xi} j \Bigg]. \end{aligned} \tag{10}$$

Notice that we only expanded $E(X + JY + j)^2$ based on the well-known form of $(a + b + c)^2$, operating with the constant j as a random variable with degenerate distribution, and using the benefit of independence between variables. The number of geometric variables follow a binomial distribution, hence the formula in the beginning of the left hand side of the equation. k in this equation is a positive integer constant, but otherwise is modelled by a Poisson distribution as in the purely exponential case above. The expectation is reduced to

$$\begin{aligned} E((X + JY + j)^2) = k\delta^2 + (j - k\delta(\zeta - 1))^2 \\ + \frac{2k\zeta J(j + \delta(k + \zeta - k\zeta - 1)) - \xi k(\delta\zeta)^2}{\xi} \\ + \frac{k\zeta J^2(2 - \xi + \zeta(k-1))}{\xi^2} \end{aligned} \tag{11}$$

The first line of the expectation is similar to the result calculated for the simple exponential shift size case, with the ζ mixing parameter appearing here. Notice however, that the result cannot be simply partitioned into a purely Erlang or a purely negative binomial part. The rest of the calculation is the same as for the exponential (non-mixture) shift size case, albeit with more complicated expressions:

$$\begin{aligned} \mathcal{C}_{h,j}^2 &= \int_0^h e^{-ts} j^2 + \sum_{k=1}^{\infty} \frac{(ts)^k e^{-ts}}{k!} \big(E((X + JY + j)^2) \big) dt \\ &= \int_0^h j^2 + \frac{2jst(\delta(\xi - \xi\zeta) + \zeta J)}{\xi} + \\ &\quad + \frac{st(\delta^2\xi^2(\zeta - 1)(-2 + st(\zeta - 1)) - 2\delta\xi\zeta st J(\zeta - 1) + \zeta(2 - \xi + \zeta st)J^2)}{\xi^2} dt \end{aligned}$$

$$= j^2 + \delta hjs - \delta hj\zeta s + \frac{hj\zeta s J}{\xi} + \\ + \frac{hs(-6\delta^2\xi^2(\zeta - 1) - 3(\xi - 2)\zeta J^2 + 2hs(\delta(\xi - \xi\zeta) + \zeta J)^2)}{6\xi^2} \quad (12)$$

The generality of the formula is highlighted by its application on mixture distributions, since there is no density function to be used to ease the calculations, like in the simpler, exponential case above. For more details and more general results, see Dobi and Zempléni (2021).

References

Anhoej J, Roeder T (2017) qicharts: quality improvement charts. https://CRAN.R-project.org/package=qicharts. R package version 0.5.5

Brooks EB, Wynne RH, Thomas VA, Blinn CE, Coulston JW (2013) On-the-fly massively multitemporal change detection using statistical quality control charts and landsat data. IEEE Trans Geosci Remote Sens 52(6):3316–3332. https://doi.org/10.1109/TGRS.2013.2272545

Buttrey SE (2009) An excel add-in for statistical process control charts. J Stat Softw 30(13):1–12. https://doi.org/10.18637/jss.v030.i13

Campitelli E (2020) metR: tools for easier analysis of meteorological fields. https://CRAN.R-project.org/package=metR. R package version 0.9.0

Correia F, Nêveda R, Oliveira P (2011) Chronic respiratory patient control by multivariate charts. Int J Health Care Qual Assur 24(8):621–643. https://doi.org/10.1108/09526861111174198

Dobi B, Zempléni A (2019a) Markov chain-based cost-optimal control charts for health care data. Qual Reliab Eng Int 35(5):1379–1395. https://doi.org/10.1002/qre.2518

Dobi B, Zempléni A (2019b) Markov chain-based cost-optimal control charts with different shift size distributions. Ann Univ Sci Budapestinensis Rolando Eötvös Nomin Sect Comput 49:129–146

Dobi B, Zempléni A (2020) Markovchart: Markov chain-based cost-optimal control charts. https://CRAN.R-project.org/package=Markovchart. R package version 2.1.4

Dobi B, Zempléni A (2021) A note on the moments of special mixture distributions, with applications for control charts. arXiv preprint 2112.05940

Duncan AJ (1956) The economic design of x charts used to maintain current control of a process. J Am Stat Assoc 51(274):228–242. https://doi.org/10.1080/01621459.1956.10501322

Flores M, Fernandez R, Naya S, Tarrio-Saavedra J (2020) qcr: quality control review. https://CRAN.R-project.org/package=qcr. R package version 1.2

Gerber F (2020) optimParallel: parallel version of the L-BFGS-B optimization method. https://CRAN.R-project.org/package=optimParallel. R package version 1.0-1

Grey K (2018) ggQC: quality control charts for 'ggplot'. https://CRAN.R-project.org/package=ggQC. R package version 0.0.31

IBM Corporation (2019) SPSS Statistics, version 26.0. Armonk, NY. www.ibm.com/products/spss-statistics

Knoth S (2020) spc: statistical process control—calculation of ARL and other control chart performance measures. https://CRAN.R-project.org/package=spc. R package version 0.6.4

McCracken A, Chakraborti S (2013) Control charts for joint monitoring of mean and variance: an overview. Qual Technol Quant Manag 10(1):17–36. https://doi.org/10.1080/16843703.2013.11673306

Meyer CD (2000) Matrix analysis and applied linear algebra. Siam, Philadelphia

Microsoft (2019) Microsoft Excel, version 16.0. Redmond, WA. https://products.office.com/en-us/excel

Montgomery DC (2009) Introduction to statistical quality control, 6th edn. Wiley, Jefferson City

Mortarino C (2010) Duncan' model for $\bar{x}$-control charts: sensitivity analysis to input parameters. Qual Reliab Eng Int 26(1):17–26. https://doi.org/10.1002/qre.1026

Phillipov G, Phillips PJ (2001) Components of total measurement error for hemoglobin a1c determination. Clin Chem 47(105):1851–1853. https://doi.org/10.1002/clc.22079

R Core Team (2020) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/

Sales RF, Vitale R, de Lima SM, Pimentel MF, Stragevitch L, Ferrer A (2016) Multivariate statistical process control charts for batch monitoring of transesterification reactions for biodiesel production based on near-infrared spectroscopy. Comput Chem Eng 94:343–353. https://doi.org/10.1016/j.compchemeng.2016.08.013

SAS Institute Inc (2013) SAS/STAT software, version 9.4. Cary, NC. https://www.sas.com/

Scrucca L, Snow G, Bloom field P (2017) qcc: quality control charts. https://CRAN.R-project.org/package=qcc. R package version 2.7

StataCorp LLC (2019) STATA, version 16. Texas, TX. https://www.stata.com/

Suman G, Prajapati D (2018) Control chart applications in healthcare: a literature review. Int J Metrol Qual Eng. https://doi.org/10.1051/ijmqe/2018003

Thor J, Lundberg J, Ask J, Olsson J, Carli C, Härenstam KP, Brommels M (2007) Application of statistical process control in healthcare improvement: systematic review. BMJ Qual Saf 16(5):387–399. https://doi.org/10.1136/qshc.2006.022194

Ton VK, Martin SS, Blumenthal RS, Blaha MJ (2013) Comparing the new European cardiovascular disease prevention guideline with prior American heart association guidelines: an editorial review. Clin Cardiol 36(5):E1–E6. https://doi.org/10.1002/clc.22079

Wallig M, Microsoft, Weston S (2020a) foreach: provides foreach looping construct. https://CRAN.R-project.org/package=foreach. R package version 1.5.1

Wallig M, Microsoft, Weston S, Tenenbaum D (2020b) doParallel: foreach parallel adaptor for the 'parallel' package. https://CRAN.R-project.org/package=doParallel. R package version 1.0.16

Wang P (2017) What clinical laboratorians should do in response to extremely low hemoglobin a1c results. Lab Med 48(1):89–92. https://doi.org/10.1093/labmed/lmw050

Wickham H, Chang W, Henry L, Pedersen TL, Takahashi K, Wilke C, Woo K, Yutani H, Dunnington D, RStudio (2020) ggplot2: create elegant data visualisations using the grammar of graphics. https://CRAN.R-project.org/package=ggplot2. R package version 3.3.3

Zempléni A, Véber M, Duarte B, Saraiva P (2004) Control charts: a cost-optimization approach for processes with random shifts. Appl Stoch Model Bus Ind 20(3):185–200. https://doi.org/10.1002/asmb.521

Zhou C, Zhang W (2015) Recurrence plot based damage detection method by integrating control chart. Entropy 17(5):2624–2641. https://doi.org/10.3390/e17052624

Zhu W, Park C (2013a) edcc: an R package for the economic design of the control chart. J Stat Softw 52(9):1–24. https://doi.org/10.18637/jss.v052.i09

Zhu W, Park C (2013b) edcc: economic design of control charts. https://CRAN.R-project.org/package=edcc. R package version 1.0-0