Projektschätzung unterstützt durch Monte Carlo Simulationen

Entwicklertag Karlsruhe 2021

Kai Gellien, GFT Deutschland

10.06.2021

Abstract

Klassische Schätzverfahren führen in der Regel zu Punktschätzungen, d.h. zum Beispiel zu einem erwarteten Liefertermin. Wenn man seine Sache etwas besser macht, so gibt man zusätzlich noch Schätzungen zum Worst- bzw. Best-Case ab. Der große Nachteil bei solchen Verfahren, gerade wenn zwischen Best- und Worst-Case Schätzung eine nennenswerte Spanne liegt, ist, dass man keinerlei Aussagen darüber erhält, wie wahrscheinlich irgendein Wert innerhalb dieses Bereichs ist. Genau dies leisten probabilistische Verfahren, wie die hier vorgestellte Monte Carlo Simulation mit Boot-Strapping. Dieser Vortrag gibt eine praktische Einführung am Beispiel eines fiktiven Projekts und zeigt insbesondere, wie überraschend einfach das Verfahren ein- und umzusetzen ist. Abschließend wird in diesem Zusammenhang noch auf #NoEstimates und auf die übliche Scrum-Praxis des Hochrechnens auf Basis der Velocity eingegangen.

Einführung

Worum geht es?

In Scrum geht man davon aus, dass sich nach ein paar Sprints die Team-Velocity einpendelt und das es auf dieser Basis, zusammen mit einem geschätzten und priorisierten Backlog möglich wird, Liefertermine zu schätzen.

Wir wollen hier an Hand eines fiktiven Projektes mehrere Möglichkeiten vorstellen, Liefertermine zu schätzen.

Schätzungen

Die Abschätzung von Zeit und Aufwand in einem Projekt ist häufig problematisch:

- Unzureichendes Wissen über den Inhalt von Arbeitspaketen
- Neuartigen Aufgaben häufig zu optimistisch geschätzt
- Unsicherheit umso größer, je größer der Zeithorizont (Cone of Uncertainty)
- Häufig trotzdem als feste „Deadline“ betrachtet
- Fazit: “Voraussagen sind schwierig, insbesondere wenn sie die Zukunft betreffen”

Welcher Philosophie folgt Euer Projekt?

- Ernst Bloch?
- Das Prinzip Hoffnung
- Hans Jonas?
- Das Prinzip Verantwortung
- Hoffnung ist keine Strategie!

Ziele des Vortrags?

- Punktschätzungen sind einfach aber suboptimal
- Vorstellung relativ einfacher Verfahren, um mehr herauszuholen.
- Vorstellung von Wahrscheinlichkeitsverteilungen
- Monte Carlo Verfahren zur Simulation

Das Ellsberg-Paradoxon

Warum wird so oft auf Punktschätzungen bestanden?

- Menschen ziehen häufig ein Risiko - z.B. in Form einer Punktschätzung - einer Situation von Ungewissheit - z.B. in Form einer Intervallschätzung - vor
- "Better be wrong and detailed than right and uncertain"

Projektschätzung - Ein Beispiel

Vorliegende Sprintergebnisse

Sprint	StoryPoints	Stories
1	10	4
2	8	2
3	17	4
4	13	3
5	13	5
	61	18

Wir haben pro Sprint durchschnittlich 12.2 StoryPoints sowie 3.6 Stories geschafft.

Backlog bis zum MVP (geschätzt und priorisiert)

Nr	StoryPoints
1	5
2	3
3	8
4	13
5	2
6	5
7	1
8	8
9	13
10	3
11	8
12	13
	82

Fragestellung

Wieviele Sprints werden benötigt, um das MVP zu realisieren?

Es soll auf Basis der bisherigen Sprintergebnisse geschätzt werden.

Ausgeklammert sind:

- Mangelnde Backlog-Qualität
- Unvollständige Stories (-> Definition-of-Ready)
- (Sehr) unterschiedliche Storygrößen
- Fluktuation im Team

Mittelwert

Es wird die mittlere Geschwindigkeit über die dokumentierten Sprints verwendet und auf die benötigten Storypoints hochgerechnet.

$$\frac{\text{StoryPoints}}{\text{StoryPoints pro Sprint}} = \frac{82}{12.2} = 6.72 \text{ Sprints (i.e. LikelyCase)}$$

Drei-Zeiten-Methode

Die Drei-Zeiten-Methode hat

$$\frac{\text{BestCase} + \text{LikelyCase} + \text{WorstCase}}{3} = \frac{4.82 + 6.72 + 10.2}{3} = 7.26 \text{ Sprints}$$

als Erwartungswert.

PERT

Gewichtung für den LikelyCase führt zu einer PERT Schätzung.

$$\frac{\text{BestCase} + 4 \times \text{LikelyCase} + \text{WorstCase}}{6} = \frac{4.82 + 4 \times 6.72 + 10.2}{6} = 6.99 \text{ Sprints}$$

Vorteile

- Einfachheit

Nachteile

- Punktschätzung, die unnötig große Genauigkeit suggeriert
  - Setzt sich in den Köpfen fest
  - Änderungen oft schwer zu vermitteln
- Information wird verschenkt bzw. unvollständig genutzt
- “Using averages, on average you’ll be wrong” (Sam Savage)

Berücksichtigung von Extrema

- Nimm den schlechtesten und den besten Sprintwert hinzu, um die Dauer nach oben bzw. unten abzuschätzen
- 8 Storypoints als schlechtestes und 17 Storypoints als bestes Sprintergebnis
- Also zwischen 10.2 und 4.82 Sprints zur Abarbeitung von 82 Storypoints benötigt
- 7.54 Sprints als Mittel aus Best- und Worst-Case
- Damit haben wir eine erste Intervallschätzung

Zwischenstand

Sprints	Verfahren
4.82	bestCase
6.72	likelyCase (Mittelwert bisheriger Sprints)
6.99	PERT
7.26	Drei Zeiten
7.54	Mittel aus best- und worstCase
10.2	worstCase

Wahrscheinlichkeitsdichtefunktion

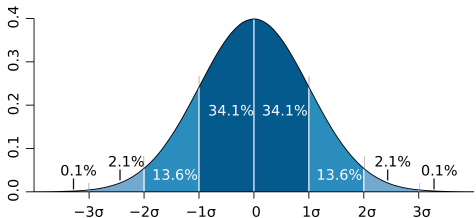


Figure 1: Dichtefunktion Normalverteilung

(Kumulative) Verteilungsfunktion

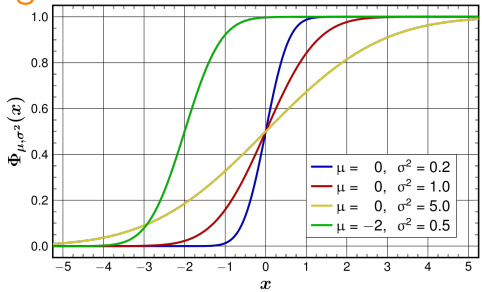


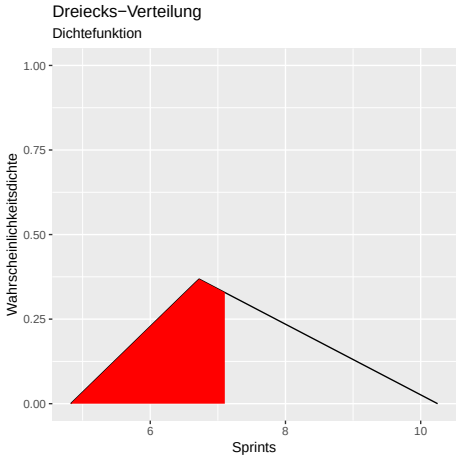
Figure 2: Verteilungsfunktion Normalverteilung

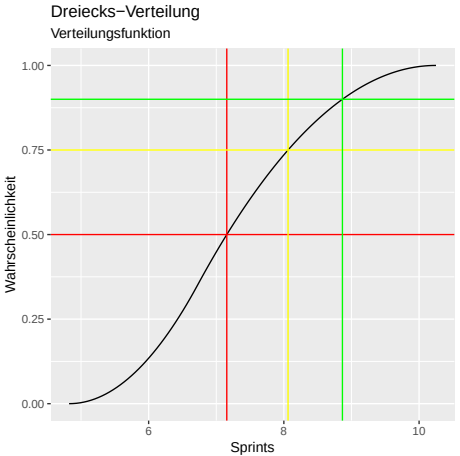
Ist X eine reelle Zufallsvariable, so nennt man die Funktion

$$F_X(x) = P(X \leq x)$$

die Verteilungsfunktion von X. Dabei bezeichnet $P(X \leq x)$ die Wahrscheinlichkeit, dass X einen Wert kleiner oder gleich x annimmt.

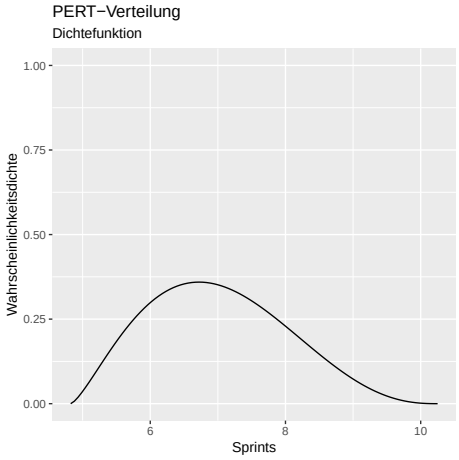
Dreiecksverteilung

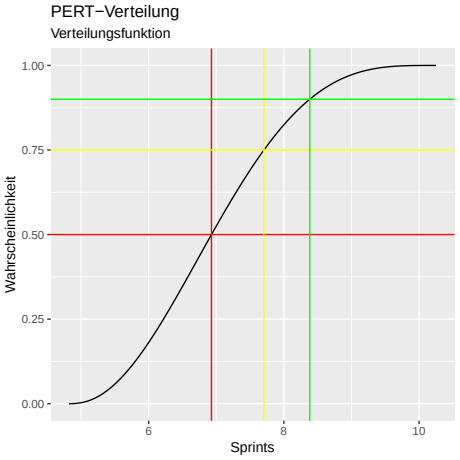




- Die Fläche unter dem Dreieck ist 1
- Der Flächeninhalt unter dieser Kurve zwischen 4.82 und 6.72 entspricht der Wahrscheinlichkeit, das Volumen innerhalb von 6.72 Sprints zu schaffen
- Wir erhalten 35%.

PERT-Verteilung





Vorteile

- Intervall- statt Punktschätzung (Max/Min)
- Nutzt Wahrscheinlichkeitsverteilung (Drei-Zeiten/PERT)

Nachteile

- Schon schwerer vermittelbar

Monte Carlo mit Bootstrap

Monte Carlo Simulation

Monte-Carlo-Simulation ist ein Verfahren aus der Stochastik, bei dem eine sehr große Zahl gleichartiger Zufallsexperimente die Basis darstellt. Es wird dabei versucht, analytisch nicht oder nur aufwendig lösbare Probleme mit Hilfe der Wahrscheinlichkeitstheorie numerisch zu lösen. Als Grundlage ist vor allem das Gesetz der großen Zahlen zu sehen.

Die Grundideen gehen auf Stanislaw Ulam zurück, der sie in den 1940er Jahren entwickelte.

Die Kreiszahl Pi wird mit der Monte-Carlo-Methode angenähert bestimmt durch das Vierfache der Wahrscheinlichkeit, mit der ein innerhalb des Quadrats zufällig gewählter Punkt in den Kreis fällt. Flächeninhalt des gesamten Kreises: πr^2, d.h. $\pi r^2/4$ für den Viertelkreis. Aufgrund des Gesetzes der großen Zahlen sinkt mit steigender Anzahl von Experimenten die Varianz des Ergebnisses.

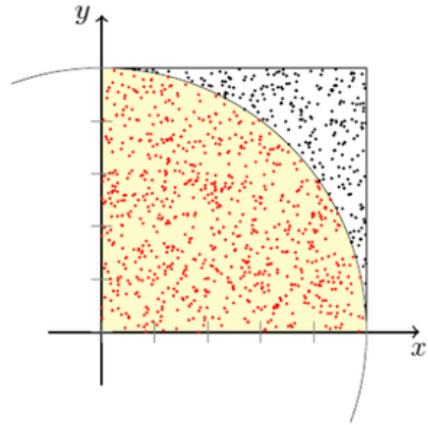
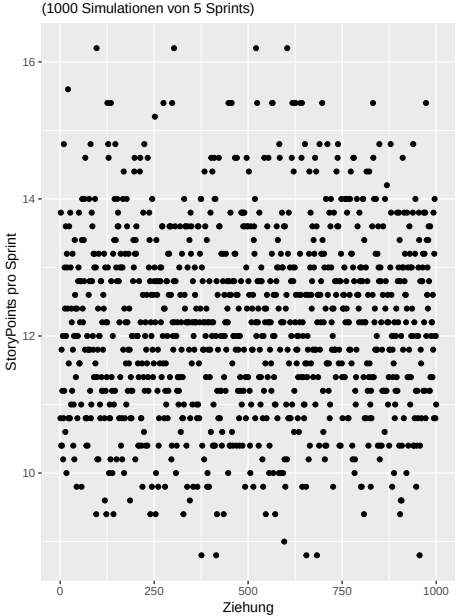
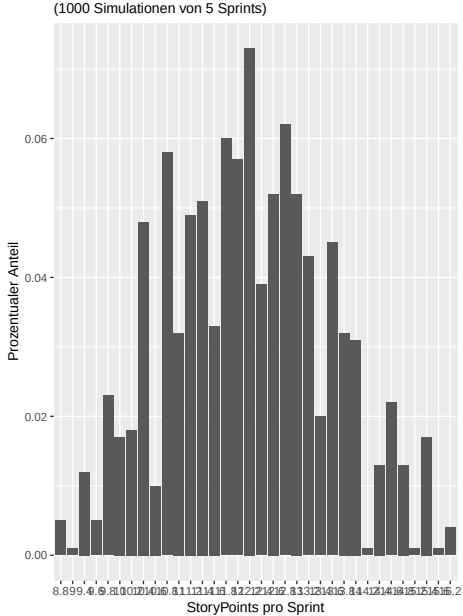


Figure 3: Berechnung von Pi per Monte Carlo Simulation

Bootstrap

Bootstrap ermöglicht, durch fortgesetztes Ziehen mit Zurücklegen aus den Ausgansdaten, eine größere Datenmenge zu generieren, die eine ähnliche Verteilung wie die zu Grunde liegende Population aufweisen wird. Verwendung finden Bootstrap-Methoden, wenn die theoretische Verteilung der interessierenden Statistik nicht bekannt ist.


Bootstrap
(1000 Simulationen von 5 Sprints)
StoryPoints pro Sprint
16
14
12
10
0
250
500
750
1000
Ziehung

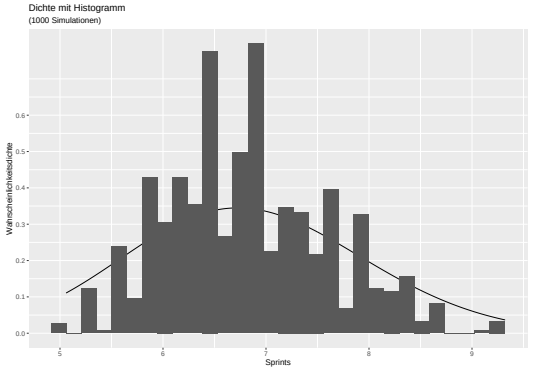

Bootstrap
(1000 Simulationen von 5 Sprints)
Prozentualer Anteil
0.06
0.04
0.02
0.00
StoryPoints pro Sprint

Nun schauen wir, wie lange wir für unsere 82 Storypoints im Backlog auf Basis obiger Sprintverläufe wohl brauchen werden. Die wesentliche Idee von Monte Carlo ist nun, dass wir Projektverläufe wie oben vielfach simulieren und dann die Verteilung der Ergebnisse betrachten.

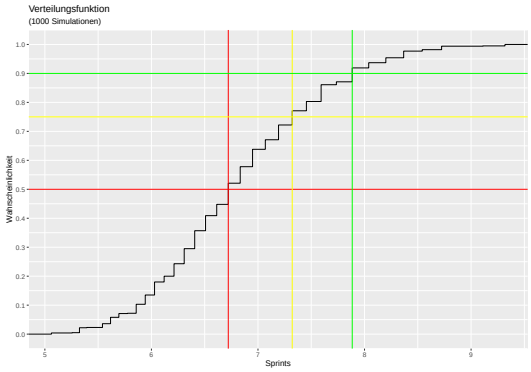
Achtung: Oben haben wir per Bootstrap schon die uns interessierende Statistik bestimmt, nämlich durchschnittliche Anzahl von Storypoints pro Sprint. Auf dieser Basis können wir direkt, gemäß Mittelwert-Methode, hochrechnen und die sich daraus ergebende Verteilung betrachten.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.062   6.308   6.721   6.821   7.321   9.318
```

Dichtefunktion mit Histogramm



Verteilungsfunktion



Aus obiger Grafik können wir nun entnehmen, dass wir in 6.72 Sprints das Ziel nur mit 50%iger Wahrscheinlichkeit erreichen können. Mit 75%iger Wahrscheinlichkeit benötigen wir 7.32 Sprints, während wir es mit 90%iger Wahrscheinlichkeit in 7.88 Sprints schaffen können.

Vorteile

- Nutzt Wahrscheinlichkeitsverteilung
- Aus (wenigen) Datenpunkten lässt sich relativ viel Information herausholen

Nachteile

- Schwerer vermittelbar
- Aufwändiger als die anderen Verfahren

Übersicht

Sprints	Verfahren
4.82	bestCase
6.72	likelyCase (Mittelwert bisheriger Sprints)
6.72	Monte Carlo 50%
6.99	PERT (Erwartungswert, d.h. 50%)
7.26	Dreieck (Erwartungswert, d.h. 50%)
7.32	Monte Carlo 75%
7.54	Mittel aus best- und worstCase
7.88	Monte Carlo 90%
10.2	worstCase

Fazit

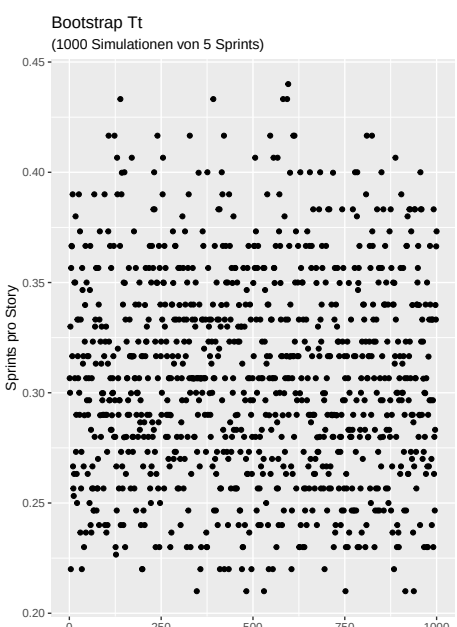
- Denke und kommuniziere in Wahrscheinlichkeiten und Verteilungen
- Schätzungen nur in angemessener Granularität!
- Grundverständnis ist wichtig, gesunder Menschenverstand auch!
- Ellsberg-Paradoxon
- Dies ist erst der Anfang ($\rightsquigarrow$ Risikomanagement)

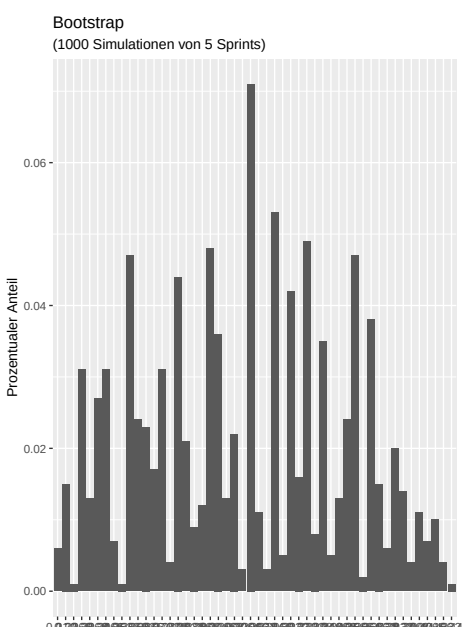
Bonus: Takt-Time

Bonus: Takt-Time

Die '#NoEstimates'-Bewegung stellt sich (vereinfacht) auf den Standpunkt, dass es reicht, auf Basis der durchschnittlich pro Sprint abgearbeiteten Stories und der Anzahl der verbleibenden Stories hochzurechnen. Also ermittelt man per Bootstrapping viele solcher Takt-Zeiten.

Analog wie oben führen wir die Simulation durch:





Betrachten wir nun eine Summary-Statistik über die simulierten Taktzeiten:

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.520   3.278   3.679   3.681   4.080   5.280
```

Eckdaten: Wir wollen 12 Stories mit insgesamt 82 Storypunkten abarbeiten, d.h wir haben im Schnitt 6.83 Storypoints pro Story. In 5 Sprints haben wir 18 Stories mit 61 Storypunkten schon geschafft. Dort hatten wir durchschnittlich 3.39 Storypoints pro Story. Im Durchschnitt haben wir damit 0.278 Sprints pro Story und 0.082 Sprints pro Storypoint benötigt. Auf Basis von Sprints pro Story kämen wir somit auf 3.33 Sprints. Auf Basis von Sprints pro Storypoint benötigten wir 6.72 Sprints.

Sprint	StoryPoints	Stories	SprintsProStory	SprintsProStoryPoint	StorypointsProStory
1	10	4	0.250	0.1000	2.50
2	8	2	0.500	0.1250	4.00
3	17	4	0.250	0.0588	4.25
4	13	3	0.333	0.0769	4.33
5	13	5	0.200	0.0769	2.60

Nr	StoryPoints
1	5
2	3
3	8
4	13
5	2
6	5
7	1
8	8
9	13
10	3
11	8
12	13
	82

Anhang

Quellen

- Dimitar Bakardzhiev: #NoEstimates Project Planning Using Monte Carlo Simulation
- Tom DeMarco & Timothy Lister - Waltzing With Bears: Managing Risk on Software Projects (Bärentango)
- Adrian Fittolani: Agile Project Forecasting – The Monte Carlo Method
- Robert C. Martin - Effective Estimation (or: How not to Lie)
- Steve McConnell - Software Estimation
- Sam Savage: The Flaw of Averages
- Gernot Starke - Raten, Schätzen, Zählen, Rechnen: Praktische Tipps zum Umgang mit Unbekanntem
- YouTube: Probability Distribution Functions (PMF, PDF, CDF)