

Strukturanalyse

Tarifrechner für Kapitallebensversicherung

Tarifrechner_KLV.xlsm

Aktuarielle Analyse für Python-Migration

January 19, 2026

Abstract

Diese Dokumentation analysiert die Struktur und Berechnungslogik des aktuariellen Tarifrechners `Tarifrechner_KLV.xlsm` für Kapitallebensversicherungen. Die Analyse dient als Grundlage für die Migration des Excel-basierten Systems nach Python. Das Dokument beschreibt die Datenstruktur, VBA-Module, versicherungsmathematischen Formeln sowie kritische Punkte für die Implementierung.

Contents

1	Übersicht	3
1.1	Dateiinformatioren	3
1.2	Produktbeschreibung	3
2	Tabellenblatt-Struktur	3
2.1	Tabellenblatt Kalkulation	3
2.1.1	Übersicht	3
2.1.2	Eingabebereich: Vertragsdaten	3
2.1.3	Eingabebereich: Tarifdaten	4
2.1.4	Eingabebereich: Grenzen für flexible Phase	4
2.1.5	Beitragsberechnungen	4
2.1.6	Verlaufswerte	5
2.2	Tabellenblatt Tafeln	6
2.2.1	Übersicht	6
2.2.2	Struktur	6
3	VBA-Modul-Struktur	7
3.1	Modul mConstants	7
3.1.1	Zweck	7
3.1.2	Konstanten	7
3.2	Modul mGWerte	7
3.2.1	Zweck	7
3.2.2	Sterbewahrscheinlichkeiten	7
3.2.3	Anzahl Lebende (l_x -Werte)	8
3.2.4	Anzahl Tote (t_x -Werte)	8
3.2.5	Kommutationswerte D_x	8
3.2.6	Kommutationswerte C_x	8
3.2.7	Kommutationswerte N_x	9
3.2.8	Kommutationswerte M_x	9
3.2.9	Kommutationswerte R_x	9
3.2.10	Altersberechnung	10
3.2.11	Cache-Management	10
3.3	Modul mBarwerte	10
3.3.1	Zweck	10
3.3.2	Lebenslange Leibrente	10
3.3.3	Temporäre Leibrente	11
3.3.4	Aufgeschobene Leibrente	11
3.3.5	Todesfallbarwert (temporär)	11
3.3.6	Erlebensfallbarwert	11
3.3.7	Endliche Rente	11
3.3.8	Abzugsglied für unterjährige Zahlungen	11
4	Definierte Namen (Named Ranges)	12
4.1	Eingabeparameter	12
4.2	Tarifdaten	12
4.3	Grenzen und berechnete Werte	12
4.4	Tafel-Bereiche	13

5	Berechnungslogik und Abhängigkeiten	13
5.1	Berechnungskette	13
5.2	Datenflussdiagramm	13
5.3	Kritische Berechnungsbestandteile	13
5.3.1	Cache-Optimierung	13
5.3.2	Array-Formeln	14
5.3.3	Rundungsgenauigkeit	14
6	Anforderungen für Python-Migration	15
6.1	Funktionale Anforderungen	15
6.1.1	Pflichtanforderungen	15
6.1.2	Optionale Anforderungen	15
6.2	Technische Anforderungen	15
6.2.1	Python-Packages	15
6.2.2	Architektur-Prinzipien	15
6.2.3	Validierungsstrategie	16
7	Kritische Punkte für Migration	16
7.1	Genauigkeitsaspekte	16
7.1.1	Rundung	16
7.1.2	Akkumulation von Rundungsfehlern	16
7.2	Performance-Aspekte	16
7.2.1	Cache-Mechanismus	16
7.2.2	Vektorisierung	16
7.3	Usability-Aspekte	17
7.3.1	Eingabe	17
7.3.2	Ausgabe	17
8	Beispielrechnung (Validierung)	17
8.1	Eingabewerte	17
8.2	Erwartete Ergebnisse (aus Excel)	17
8.2.1	Beitragsberechnung	17
8.2.2	Verlaufswerte für $k = 0$	17
8.3	Validierungsstrategie	17
9	Zusammenfassung	18
9.1	Kernpunkte	18
9.2	Nächste Schritte für Python-Migration	18
9.3	Kontaktinformationen	19

1 Übersicht

1.1 Dateiinformationen

- **Datei:** Tarifrechner_KLV.xlsm
- **Typ:** Excel Macro-Enabled Workbook
- **Zweck:** Aktuarieller Tarifrechner für Kapitallebensversicherung (KLV)
- **Berechnungsbasis:** DAV-Sterbetafeln (Deutsche Aktuarvereinigung)
- **Tafeln:** DAV1994_T, DAV2008_T (jeweils männlich/weiblich)

1.2 Produktbeschreibung

Der Tarifrechner berechnet Beiträge und Deckungsrückstellungen für eine Kapitallebensversicherung mit folgenden Charakteristika:

- Kombinierte Todes- und Erlebensfalleistung
- Temporäre Beitragszahlung (Dauer $t \leq n$)
- Versicherungsdauer n Jahre
- Unterjährige Beitragszahlung möglich (jährlich, halbjährlich, vierteljährlich, monatlich)
- Kostenbelastung: Abschluss-, Inkasso- und Verwaltungskosten

2 Tabellenblatt-Struktur

2.1 Tabellenblatt Kalkulation

2.1.1 Übersicht

- **Dimensionen:** A1:L66
- **Maximale Zeile:** 66
- **Maximale Spalte:** 12 (A–L)
- **Anzahl Formeln:** 309
- **Typ:** Berechnungsblatt mit Eingaben und Ergebnissen

2.1.2 Eingabebereich: Vertragsdaten

Spalten A–B, Zeilen 4–9:

Zulässige Werte für Zahlungsweise (zw):

- $zw = 1$: jährliche Zahlungsweise
- $zw = 2$: halbjährliche Zahlungsweise
- $zw = 4$: vierteljährliche Zahlungsweise
- $zw = 12$: monatliche Zahlungsweise

Parameter	Symbol	Beispielwert
Eintrittsalter	x	40 Jahre
Geschlecht	Sex	M (männlich)
Versicherungsdauer	n	30 Jahre
Beitragszahlungsdauer	t	20 Jahre
Versicherungssumme	VS	100.000 EUR
Zahlungsweise	zw	12 (monatlich)

Table 1: Vertragsdaten – Eingabeparameter

Parameter	Symbol	Beispielwert
Rechnungszins	i	1,75%
Sterbetafel	Tafel	DAV1994_T
Abschlusskostensatz	α	2,50%
Inkassokostensatz	β_1	2,50%
Verwaltungskostensatz Tod	γ_1	0,08%
Verwaltungskostensatz Erlebensfall	γ_2	0,125%
Verwaltungskostensatz DR	γ_3	0,25%
Stückkosten	k	24,00 EUR
Ratenzuschlag	ratzu	5%

Table 2: Tarifdaten – Kostensätze und Parameter

2.1.3 Eingabebereich: Tarifdaten

Spalten D–E, Zeilen 4–12:

Hinweis zu Stückkosten: Der Parameter k (Zelle E11) bezeichnet die Stückkosten in EUR, nicht die Anzahl Zahlungen pro Jahr.

Ratenzuschlag: Wird per Formel aus Zahlungsweise berechnet:

$$\text{ratzu} = \begin{cases} 2\% & \text{falls } zw = 2 \\ 3\% & \text{falls } zw = 4 \\ 5\% & \text{falls } zw = 12 \\ 0\% & \text{sonst} \end{cases} \quad (1)$$

2.1.4 Eingabebereich: Grenzen für flexible Phase

Spalten G–H, Zeilen 4–5:

Parameter	Symbol	Wert
Mindestalter für flexible Phase	MinAlterFlex	60 Jahre
Mindestrestlaufzeit für flexible Phase	MinRLZFlex	5 Jahre

Table 3: Grenzen für flexible Phase

2.1.5 Beitragsberechnungen

Spalten J–K, Zeilen 5–9:

Array-Formel für Bruttobeitrag Tod (Zelle K5):

Größe	Formel	Zelle
Bruttobeitrag Tod (je 1 EUR VS)	$B_{x:t}$	K5 (Array-Formel)
Jahresbeitrag	BJB	K6: $= VS \cdot B_{x:t}$
Zahlbeitrag	BZB	K7: $= \frac{1+\text{rat zu}}{zw} \cdot (BJB + k)$
Nettoprämiensatz Tod	$P_{x:t}$	K9 (Array-Formel)

Table 4: Beitragsberechnungen

$$B_{x:t} = \frac{{}_nA_x + {}_nE_x + \gamma_1 \cdot \ddot{a}_{x:t}^{(1)} + \gamma_2 \cdot (\ddot{a}_{x:n}^{(1)} - \ddot{a}_{x:t}^{(1)})}{(1 - \beta_1) \cdot \ddot{a}_{x:t}^{(1)} - \alpha \cdot t} \quad (2)$$

wobei:

- ${}_nA_x$ = Barwert temporäre Todesfallversicherung
- ${}_nE_x$ = Barwert reine Erlebensfallversicherung
- $\ddot{a}_{x:t}^{(1)}$ = Barwert temporäre vorschüssige Leibrente (jährlich)
- $\ddot{a}_{x:n}^{(1)}$ = Barwert temporäre vorschüssige Leibrente (jährlich, n Jahre)

Array-Formel für Nettoprämiensatz Tod (Zelle K9):

$$P_{x:t} = \frac{{}_nA_x + {}_nE_x + t \cdot \alpha \cdot B_{x:t}}{\ddot{a}_{x:t}^{(1)}} \quad (3)$$

2.1.6 Verlaufswerte

Zeilen 14–66: Berechnung versicherungsmathematischer Größen für jedes Vertragsjahr k .
Spaltenüberschriften (Zeile 15):

Spalte	Bezeichnung	Beschreibung
A	k	Vertragsjahr
B	${}_{n-k}A_{x+k}$	Todesfallbarwert ab Jahr k
C	$\ddot{a}_{x+k:n-k}^{(1)}$	Rentenbarwert gesamt ab Jahr k
D	$\ddot{a}_{x+k:t-k}^{(1)}$	Rentenbarwert Beitragszahlung ab Jahr k
E	${}_kV_x^{\text{bpfl}}$	Deckungsrückstellung beitragspflichtig (je 1 EUR VS)
F	${}_kDR_x^{\text{bpfl}}$	Deckungsrückstellung beitragspflichtig (EUR)
G	${}_kV_x^{\text{bfr}}$	Deckungsrückstellung beitragsfrei (je 1 EUR VS)
H	${}_kV_x^{\text{MRV}}$	Deckungsrückstellung Mindestrückkaufswert (je 1 EUR VS)
I	–	Flexible Phase (Kennzeichen)
J	StoAb	Stornoabzug
K	RKW	Rückkaufswert
L	VS_{bfr}	Versicherungssumme beitragsfrei

Table 5: Verlaufswerte – Spaltenüberschriften

Zentrale Array-Formeln für Verlaufswerte:

$${}_{n-k}A_{x+k} = \begin{cases} \frac{M_{x+k}-M_{x+n}}{D_{x+k}} + \frac{D_{x+n}}{D_{x+k}} & \text{falls } k \leq n \\ 0 & \text{sonst} \end{cases} \quad (4)$$

$$\ddot{a}_{x+k:n-k}^{(1)} = \frac{N_{x+k} - N_{x+n}}{D_{x+k}} - \beta(1, i) \cdot \left(1 - \frac{D_{x+n}}{D_{x+k}}\right) \quad (5)$$

$$\ddot{a}_{x+k:t-k}^{(1)} = \frac{N_{x+k} - N_{x+t}}{D_{x+k}} - \beta(1, i) \cdot \left(1 - \frac{D_{x+t}}{D_{x+k}}\right) \quad (6)$$

Deckungsrückstellung beitragspflichtig:

$${}_kV_x^{\text{bpfl}} = {}_{n-k}A_{x+k} - P_{x:t} \cdot \ddot{a}_{x+k:t-k}^{(1)} + \gamma_2 \cdot \left(\ddot{a}_{x+k:n-k}^{(1)} - \frac{\ddot{a}_{x:n}^{(1)}}{\ddot{a}_{x:t}^{(1)}} \cdot \ddot{a}_{x+k:t-k}^{(1)} \right) \quad (7)$$

Deckungsrückstellung beitragsfrei:

$${}_kV_x^{\text{bfr}} = {}_{n-k}A_{x+k} + \gamma_3 \cdot \ddot{a}_{x+k:n-k}^{(1)} \quad (8)$$

2.2 Tabellenblatt Tafeln

2.2.1 Übersicht

- **Dimensionen:** A3:E127
- **Maximale Zeile:** 127
- **Maximale Spalte:** 5 (A–E)
- **Anzahl Formeln:** 0
- **Typ:** Reines Datenblatt (Sterbewahrscheinlichkeiten)

2.2.2 Struktur

Header (Zeile 3):

Spalte	Bezeichnung
A	x/y (Alter)
B	DAV1994_T_M (DAV 1994, männlich)
C	DAV1994_T_F (DAV 1994, weiblich)
D	DAV2008_T_M (DAV 2008, männlich)
E	DAV2008_T_F (DAV 2008, weiblich)

Table 6: Tafeln – Spaltenstruktur

Daten (Zeilen 4–127):

- Alter: $x \in \{0, 1, 2, \dots, 123\}$
- Sterbewahrscheinlichkeiten q_x für jede Tafel
- Letztes Alter ($x = 123$): $q_{123} = 1,0$ (sicherer Tod)

Beispiel-Daten:

x	DAV1994_T_M	DAV1994_T_F	DAV2008_T_M	DAV2008_T_F
0	0,011687	0,009003	0,006113	0,005088
40	0,002378	0,001233	0,001618	0,000680
60	0,007891	0,003932	0,005913	0,002861
123	1,000000	1,000000	1,000000	1,000000

Table 7: Beispiel-Sterbewahrscheinlichkeiten

3 VBA-Modul-Struktur

3.1 Modul mConstants

3.1.1 Zweck

Definiert globale Konstanten für Rundung und Altersgrenzen.

3.1.2 Konstanten

```

1 Public Const rund_lx As Integer = 16 ' Rundung fuer Lebende
2 Public Const rund_tx As Integer = 16 ' Rundung fuer Tote
3 Public Const rund_Dx As Integer = 16 ' Rundung fuer Dx-Werte
4 Public Const rund_Cx As Integer = 16 ' Rundung fuer Cx-Werte
5 Public Const rund_Nx As Integer = 16 ' Rundung fuer Nx-Werte
6 Public Const rund_Mx As Integer = 16 ' Rundung fuer Mx-Werte
7 Public Const rund_Rx As Integer = 16 ' Rundung fuer Rx-Werte
8 Public Const max_Alter As Integer = 123 ' Maximales Alter

```

Listing 1: VBA-Code: Modul mConstants

Erläuterung:

- Alle Rundungen auf 16 Dezimalstellen
- Höchstalter 123 Jahre (entspricht Tafel-Maximum)
- Verwendung von `WorksheetFunction.Round` mit Banker's Rounding

3.2 Modul mGWerte

3.2.1 Zweck

Berechnung aktuarieller Grundwerte (Commutation Functions).

3.2.2 Sterbewahrscheinlichkeiten

Funktion: `Act_qx(Alter, Sex, Tafel, ...)`

- Liest Sterbewahrscheinlichkeit aus Tabellenblatt Tafeln
- Unterstützte Tafeln: DAV1994_T, DAV2008_T
- Geschlecht: M (männlich), F (weiblich)
- Gibt q_x -Wert zurück

3.2.3 Anzahl Lebende (l_x -Werte)

Rekursive Berechnung:

$$l_0 = 1\,000\,000 \quad (9)$$

$$l_{x+1} = l_x \cdot (1 - q_x) \quad \text{für } x = 0, 1, \dots, 122 \quad (10)$$

Nach jeder Berechnung: Rundung auf `rund_1x` = 16 Dezimalstellen.

Funktionen:

- `v_1x(Endalter, Sex, Tafel, ...)` – Private Funktion, erzeugt Vektor
- `Act_1x(Alter, Sex, Tafel, ...)` – Public Funktion, gibt Einzelwert zurück

3.2.4 Anzahl Tote (t_x -Werte)

Berechnung:

$$t_x = l_x - l_{x+1} \quad (11)$$

Funktionen:

- `v_tx(Endalter, Sex, Tafel, ...)` – Private Funktion, erzeugt Vektor
- `Act_tx(Alter, Sex, Tafel, ...)` – Public Funktion, gibt Einzelwert zurück

3.2.5 Kommutationswerte D_x

Berechnung:

$$D_x = l_x \cdot v^x \quad \text{mit } v = \frac{1}{1+i} \quad (12)$$

wobei v der Diskontierungsfaktor und i der Rechnungszins ist.

Funktionen:

- `v_Dx(Endalter, Sex, Tafel, Zins, ...)` – Private Funktion, erzeugt Vektor
- `Act_Dx(Alter, Sex, Tafel, Zins, ...)` – Public Funktion mit Cache

Cache-Mechanismus:

- Prüft Dictionary-Cache vor Berechnung
- Speichert Ergebnis im Cache
- Cache-Key: `Dx_Alter_Sex_Tafel_Zins_...`

3.2.6 Kommutationswerte C_x

Berechnung:

$$C_x = t_x \cdot v^{x+1} \quad (13)$$

Funktionen:

- `v_Cx(Endalter, Sex, Tafel, Zins, ...)` – Private Funktion, erzeugt Vektor
- `Act_Cx(Alter, Sex, Tafel, Zins, ...)` – Public Funktion mit Cache

3.2.7 Kommutationswerte N_x

Berechnung (rückwärts):

$$N_\omega = D_\omega \quad \text{mit } \omega = 123 \quad (14)$$

$$N_x = N_{x+1} + D_x \quad \text{für } x = \omega - 1, \omega - 2, \dots, 0 \quad (15)$$

Mathematisch äquivalent zu:

$$N_x = \sum_{j=x}^{\omega} D_j \quad (16)$$

Funktionen:

- `v_Nx(Sex, Tafel, Zins, ...)` – Private Funktion, erzeugt Vektor
- `Act_Nx(Alter, Sex, Tafel, Zins, ...)` – Public Funktion mit Cache

3.2.8 Kommutationswerte M_x

Berechnung (analog zu N_x):

$$M_\omega = C_\omega \quad (17)$$

$$M_x = M_{x+1} + C_x \quad (18)$$

Mathematisch:

$$M_x = \sum_{j=x}^{\omega} C_j \quad (19)$$

Funktionen:

- `v_Mx(Sex, Tafel, Zins, ...)` – Private Funktion, erzeugt Vektor
- `Act_Mx(Alter, Sex, Tafel, Zins, ...)` – Public Funktion mit Cache

3.2.9 Kommutationswerte R_x

Berechnung (analog zu N_x):

$$R_\omega = M_\omega \quad (20)$$

$$R_x = R_{x+1} + M_x \quad (21)$$

Mathematisch:

$$R_x = \sum_{j=x}^{\omega} M_j \quad (22)$$

Funktionen:

- `v_Rx(Sex, Tafel, Zins, ...)` – Private Funktion, erzeugt Vektor
- `Act_Rx(Alter, Sex, Tafel, Zins, ...)` – Public Funktion mit Cache

3.2.10 Altersberechnung

Funktion: Act_Altersberechnung(GebDat, BerDat, Methode)

- Methode K: Kalenderjahresmethode

$$\text{Alter} = \text{Jahr}_{\text{Ber}} - \text{Jahr}_{\text{Geb}} \quad (23)$$

- Methode H: Halbjahresmethode (Standard)

$$\text{Alter} = \left\lfloor \text{Jahr}_{\text{Ber}} - \text{Jahr}_{\text{Geb}} + \frac{1}{12} \cdot (\text{Monat}_{\text{Ber}} - \text{Monat}_{\text{Geb}} + 5) \right\rfloor \quad (24)$$

3.2.11 Cache-Management

Initialisierung:

```

1 Sub InitializeCache()
2     Set cache = CreateObject("Scripting.Dictionary")
3 End Sub

```

Listing 2: Cache-Initialisierung

Cache-Key-Generierung:

```

1 Private Function CreateCacheKey(Art As String, Alter As Integer, _
2     Sex As String, Tafel As String, _
3     Zins As Double, GebJahr As Integer, _
4     Rentenbeginnalter As Integer, _
5     Schicht As Integer) As String
6     CreateCacheKey = Art & "_" & Alter & "_" & Sex & "_" & _
7     Tafel & "_" & Zins & "_" & GebJahr & "_" & _
8     Rentenbeginnalter & "_" & Schicht
9 End Function

```

Listing 3: Cache-Key-Funktion

Cache-Strategie:

- Vermeidet Mehrfachberechnung identischer Werte
- Wichtig bei Array-Formeln mit vielen Funktionsaufrufen
- Cache existiert nur während Excel-Sitzung

3.3 Modul mBarwerte

3.3.1 Zweck

Berechnung versicherungsmathematischer Barwerte (Leibrenten, Versicherungen).

3.3.2 Lebenslange Leibrente

Funktion: Act_ax_k(Alter, Sex, Tafel, Zins, k, ...)

Berechnet Barwert einer lebenslangen vorschüssigen Leibrente mit k Zahlungen pro Jahr.

Formel:

$$\ddot{a}_x^{(k)} = \frac{N_x}{D_x} - \beta(k, i) \quad (25)$$

wobei $\beta(k, i)$ das Abzugsglied für unterjährige Zahlungen ist.

3.3.3 Temporäre Leibrente

Funktion: Act_axn_k(Alter, n, Sex, Tafel, Zins, k, ...)

Berechnet Barwert einer temporären vorschüssigen Leibrente (n Jahre, k Zahlungen pro Jahr).

Formel:

$$\ddot{a}_{x:\overline{n}}^{(k)} = \frac{N_x - N_{x+n}}{D_x} - \beta(k, i) \cdot \left(1 - \frac{D_{x+n}}{D_x}\right) \quad (26)$$

3.3.4 Aufgeschobene Leibrente

Funktion: Act_nax_k(Alter, n, Sex, Tafel, Zins, k, ...)

Berechnet Barwert einer aufgeschobenen vorschüssigen Leibrente (Beginn nach n Jahren).

Formel:

$${}_n\ddot{a}_x^{(k)} = \frac{D_{x+n}}{D_x} \cdot \ddot{a}_{x+n}^{(k)} \quad (27)$$

3.3.5 Todesfallbarwert (temporär)

Funktion: Act_nGrAx(Alter, n, Sex, Tafel, Zins, ...)

Berechnet Barwert einer temporären Todesfallversicherung (Leistung am Jahresende).

Formel:

$${}_nA_x = \frac{M_x - M_{x+n}}{D_x} \quad (28)$$

3.3.6 Erlebensfallbarwert

Funktion: Act_nGrEx(Alter, n, Sex, Tafel, Zins, ...)

Berechnet Barwert einer reinen Erlebensfallversicherung (Leistung nach n Jahren).

Formel:

$${}_nE_x = \frac{D_{x+n}}{D_x} \quad (29)$$

3.3.7 Endliche Rente

Funktion: Act_ag_k(g, Zins, k)

Berechnet Barwert einer endlichen vorschüssigen Rente (g Zahlungen, kein Todesfallrisiko).

Formel:

$$\ddot{a}_g^{(k)} = \begin{cases} \frac{1-v^g}{1-v} - \beta(k, i) \cdot (1-v^g) & \text{falls } i > 0 \\ g & \text{falls } i = 0 \end{cases} \quad (30)$$

3.3.8 Abzugsglied für unterjährige Zahlungen

Funktion: Act_Abzugsglied(k, Zins)

Korrektur für unterjährige Rentenzahlungen nach Woolhouse-Näherung (1. Ordnung).

Formel:

$$\beta(k, i) = \frac{1+i}{k} \sum_{\ell=0}^{k-1} \frac{\ell/k}{1 + (\ell/k) \cdot i} \quad (31)$$

Für $k = 1$ (jährliche Zahlungen): $\beta(1, i) = 0$.

Vereinfachte Woolhouse-Formel:

Für praktische Anwendungen oft genähert als:

$$\beta(k, i) \approx \frac{k-1}{2k} \quad (32)$$

Diese Näherung ist für kleine Zinssätze ausreichend genau.

4 Definierte Namen (Named Ranges)

Die Excel-Arbeitsmappe verwendet 24 definierte Namen für zentrale Parameter und Bereiche.

4.1 Eingabeparameter

Name	Zellbezug	Beschreibung
x	Kalkulation!\$B\$4	Eintrittsalter
Sex	Kalkulation!\$B\$5	Geschlecht (M/F)
n	Kalkulation!\$B\$6	Versicherungsdauer
t	Kalkulation!\$B\$7	Beitragszahlungsdauer
VS	Kalkulation!\$B\$8	Versicherungssumme
zw	Kalkulation!\$B\$9	Zahlungsweise

Table 8: Definierte Namen – Eingabeparameter Vertragsdaten

4.2 Tarifdaten

Name	Zellbezug	Beschreibung
Zins	Kalkulation!\$E\$4	Rechnungszins i
Tafel	Kalkulation!\$E\$5	Sterbetafel
alpha	Kalkulation!\$E\$6	Abschlusskostensatz α
beta1	Kalkulation!\$E\$7	Inkassokostensatz β_1
gamma1	Kalkulation!\$E\$8	Verwaltungskostensatz Tod γ_1
gamma2	Kalkulation!\$E\$9	Verwaltungskostensatz Erlebensfall γ_2
gamma3	Kalkulation!\$E\$10	Verwaltungskostensatz DR γ_3
k	Kalkulation!\$E\$11	Stückkosten
ratzu	Kalkulation!\$E\$12	Ratenzuschlag

Table 9: Definierte Namen – Tarifdaten

4.3 Grenzen und berechnete Werte

Name	Zellbezug	Beschreibung
MinAlterFlex	Kalkulation!\$H\$4	Mindestalter flexible Phase
MinRLZFlex	Kalkulation!\$H\$5	Mindestrestlaufzeit flexible Phase
B_xt	Kalkulation!\$K\$5	Bruttobeitrag Tod $B_{x:t}$
BJB	Kalkulation!\$K\$6	Jahresbeitrag
P_xt	Kalkulation!\$K\$9	Nettoprämienatz Tod $P_{x:t}$

Table 10: Definierte Namen – Grenzen und Berechnungen

Name	Zellbezug	Beschreibung
v_Tafeln	Tafeln!\$B\$3:\$E\$3	Tafelbezeichnungen (Header)
m_Tafeln	Tafeln!\$B\$4:\$E\$127	q_x -Werte (Matrix)
v_x	Tafeln!\$A\$4:\$A\$127	Alter $x = 0, \dots, 123$

Table 11: Definierte Namen – Tafel-Bereiche

4.4 Tafel-Bereiche

5 Berechnungslogik und Abhängigkeiten

5.1 Berechnungskette

Die Berechnung erfolgt in folgender hierarchischer Reihenfolge:

1. **Eingabe:** Vertragsdaten (x , Sex, n , t , VS, zw) und Tarifdaten (i , Tafel, Kostensätze)
2. **VBA mGWerte:** Grundwerte
 - q_x -Werte aus Sterbetafeln
 - l_x (Überlebende): $l_0 = 1\,000\,000$, $l_{x+1} = l_x \cdot (1 - q_x)$
 - Kommutationswerte: D_x , C_x , N_x , M_x , R_x
 - Mit Cache-Mechanismus
3. **VBA mBarwerte:** Barwerte
 - Leibrenten: $\ddot{a}_x^{(k)}$, $\ddot{a}_{x:\overline{n}}^{(k)}$
 - Versicherungsbarwerte: ${}_nA_x$, ${}_nE_x$
4. **Excel-Formel K5:** Beitrag $B_{x:t}$
 - Zähler: Todesfallbarwert + Erlebensfallbarwert + Kostenzuschläge
 - Nenner: Beitragszahlungsbarwert abzüglich Abschlusskosten
5. **Excel-Formeln K6–K7:** BJB, BZB (mit Ratenzuschlag)
6. **Excel-Formel K9:** Nettoprämiensatz $P_{x:t}$
7. **Excel-Formeln Zeilen 16+:** Verlaufswerte pro Vertragsjahr
 - Barwerte: ${}_{n-k}A_{x+k}$, $\ddot{a}_{x+k:n-k}^{(1)}$, $\ddot{a}_{x+k:t-k}^{(1)}$
 - Deckungsrückstellungen: ${}_kV_x^{\text{bpf}}$, ${}_kV_x^{\text{bfr}}$
 - Rückkaufswerte: RKW

5.2 Datenflussdiagramm

5.3 Kritische Berechnungsbestandteile

5.3.1 Cache-Optimierung

- D_x , C_x , N_x , M_x , R_x werden gecacht
- Vermeidet Mehrfachberechnung bei Array-Formeln
- Cache-Invalidierung: Nur bei Excel-Neustart
- Wichtig für Performance bei großen Verlaufstabellen

EINGABE: Vertragsdaten & Tarifdaten (Kalkulation)

v

STERBETAFELN: qx-Werte (Tafeln B4:E127)

v

VBA mG Werte: Berechnung Grundwerte

- lx (Überlebende)
- Dx, Cx, Nx, Mx, Rx (Kommutionswerte)
- Mit Cache-Mechanismus

v

VBA mBarwerte: Berechnung Barwerte

- ax_k, axn_k (Leibrenten)
- nGrAx (Todesfallbarwert)
- nGrEx (Erlebensfallbarwert)

v

EXCEL-FORMELN: Beitragsberechnung

- K5: Bxt (Bruttobeitrag Tod)
- K9: Pxt (Nettoprämienatz Tod)

v

EXCEL-FORMELN: Verlaufswerte (Zeilen 16+)

- Barwerte pro Vertragsjahr
- Deckungsrückstellungen
- Rückkaufswerte

Figure 1: Datenfluss der Tarifberechnung

5.3.2 Array-Formeln

- K5, K9: Berechnungsintensive Hauptformeln
- Spalten B–E (Zeilen 16+): Array-Formeln für Verlaufswerte
- **Vorteil:** Kompakte Darstellung, automatische Aktualisierung
- **Nachteil:** Langsam bei vielen Zeilen, schwer zu debuggen

5.3.3 Rundungsgenauigkeit

- Alle VBA-Berechnungen: 16 Dezimalstellen
- Excel-Anzeige: Oft weniger Stellen (nur Formatierung)
- **Kritisch:** Kleine Abweichungen können sich akkumulieren
- **Banker's Rounding:** Excel verwendet kaufmännische Rundung

6 Anforderungen für Python-Migration

6.1 Funktionale Anforderungen

6.1.1 Pflichtenanforderungen

1. Berechnung identischer Ergebnisse wie Excel-Version (Toleranz $< 10^{-6}$)
2. Unterstützung DAV1994_T und DAV2008_T Tafeln (männlich/weiblich)
3. Alle Barwert-Funktionen:
 - Leibrenten: $\ddot{a}_x^{(k)}$, $\ddot{a}_{x:\overline{n}}^{(k)}$, $n\ddot{a}_x^{(k)}$
 - Versicherungen: nA_x , nE_x
 - Abzugsglied: $\beta(k, i)$
4. Beitragsberechnung: $B_{x:t}$, $P_{x:t}$
5. Verlaufswerte für beliebige Anzahl Vertragsjahre
6. Validierung zulässiger Zahlungsweisen: $zw \in \{1, 2, 4, 12\}$

6.1.2 Optionale Anforderungen

1. Caching-Mechanismus für Performance-Optimierung
2. Validierung aller Eingabeparameter (Wertebereich, Konsistenz)
3. Export nach Excel/PDF
4. Grafische Darstellung der Verlaufswerte
5. Batch-Berechnungen für Parametervariationen

6.2 Technische Anforderungen

6.2.1 Python-Packages

Package	Verwendungszweck
numpy	Array-Operationen, Vektorisierung, mathematische Funktionen
pandas	Datenmanagement, Tafel-Handling, DataFrame-Export
openpyxl	Excel-Import/Export (optional)
matplotlib	Visualisierung (optional)

Table 12: Erforderliche Python-Packages

6.2.2 Architektur-Prinzipien

1. **Modularer Aufbau:** Analog zu VBA-Modulen
 - Modul `constants.py`: Konstanten
 - Modul `grundwerte.py`: Kommutationswerte
 - Modul `barwerte.py`: Leibrenten und Versicherungen
 - Modul `tarif.py`: Beitragsberechnung

2. **Klare Trennung:** Daten – Logik – Präsentation
3. **Testbarkeit:** Unit-Tests für alle Funktionen
4. **Dokumentation:** Docstrings nach NumPy-Konvention

6.2.3 Validierungsstrategie

1. Vergleich Python vs. Excel für Standardtestfälle
2. Toleranz für Rundungsdifferenzen: $\epsilon = 10^{-10}$ (absolut)
3. Relative Toleranz bei großen Werten: 10^{-6}
4. Test-Suite mit verschiedenen Parameterkombinationen

7 Kritische Punkte für Migration

7.1 Genauigkeitsaspekte

7.1.1 Rundung

- **Excel:** `WorksheetFunction.Round` verwendet Banker's Rounding
- **Python:** `numpy.round` verwendet dieselbe Methode (Standard)
- **Kritisch:** Konsistenz prüfen bei Grenzfällen (z. B. $0,5 \rightarrow$ nächste gerade Zahl)

7.1.2 Akkumulation von Rundungsfehlern

- **l_x -Werte:** Rekursive Berechnung akkumuliert Rundungsfehler
- **N_x, M_x, R_x :** Summationen vergrößern Abweichungen
- **Lösung:** Hohe Präzision beibehalten (`np.float64`)
- **Rundung:** Erst am Ende der Berechnungskette

7.2 Performance-Aspekte

7.2.1 Cache-Mechanismus

- **Excel:** Dictionary in VBA, Lebensdauer = Sitzung
- **Python:**
 - Decorator `@lru_cache` aus `functools`
 - Oder dict-basierter Cache in Klasse
- **Wichtig:** Cache-Keys müssen alle Parameter beinhalten
- **Hashable:** Nur immutable Typen als Keys verwenden

7.2.2 Vektorisierung

- **Vermeidung von Schleifen:** Nutze NumPy-Array-Operationen
- **Vectorize:** `numpy.vectorize` oder direkte Array-Operationen
- **Profiling:** Mit `timeit` oder `cProfile`
- **Beispiel:** Berechnung aller l_x auf einmal statt Schleife

7.3 Usability-Aspekte

7.3.1 Eingabe

- **Excel:** Direkte Eingabe in Zellen
- **Python:**
 - Parameterübergabe an Funktionen
 - Config-File (JSON, YAML)
 - Kommandozeilen-Interface (CLI)
 - GUI (z. B. mit Streamlit)

7.3.2 Ausgabe

- **Excel:** Tabelle mit Verlaufswerten
- **Python:**
 - `pandas.DataFrame` für Verlaufswerte
 - Excel-Export mit `openpyxl`
 - Grafiken mit `matplotlib`
 - PDF-Report mit `reportlab`

8 Beispielrechnung (Validierung)

8.1 Eingabewerte

Parameter	Symbol	Wert
Eintrittsalter	x	40 Jahre
Geschlecht	Sex	M (männlich)
Versicherungsdauer	n	30 Jahre
Beitragszahlungsdauer	t	20 Jahre
Versicherungssumme	VS	100.000 EUR
Zahlungsweise	zw	12 (monatlich)
Rechnungszins	i	1,75%
Sterbetafel	Tafel	DAV1994_T

Table 13: Beispielrechnung – Eingabeparameter

8.2 Erwartete Ergebnisse (aus Excel)

8.2.1 Beitragsberechnung

8.2.2 Verlaufswerte für $k = 0$

8.3 Validierungsstrategie

1. Berechne mit Python-Implementation
2. Vergleiche mit Excel-Werten
3. Akzeptiere Abweichung:

Größe	Wert	Einheit
$B_{x:t}$ (K5)	0,04226001	je 1 EUR VS
BJB (K6)	4.226,00	EUR
BZB (K7)	371,88	EUR (monatlich)
$P_{x:t}$ (K9)	0,04001217	je 1 EUR VS

Table 14: Beispielrechnung – Beitragsergebnisse

Größe	Wert	Einheit
${}_{30}A_{40}$	0,6315923	je 1 EUR VS
$\ddot{a}_{40:30}^{(1)}$	21,4202775	–
$\ddot{a}_{40:20}^{(1)}$	16,3130941	–
${}_0V_{40}^{\text{bpf}}$	–0,0211300	je 1 EUR VS

Table 15: Beispielrechnung – Verlaufswerte zu Vertragsbeginn

- Absolute Toleranz: $|\text{Python} - \text{Excel}| < 10^{-6}$
- Relative Toleranz: $\frac{|\text{Python} - \text{Excel}|}{|\text{Excel}|} < 10^{-6}$

4. Bei Abweichung: Analyse der Ursache (Rundung, Formel, Cache)

9 Zusammenfassung

9.1 Kernpunkte

- Der Tarifrechner besteht aus 2 Tabellenblättern und 3 VBA-Modulen
- Berechnung basiert auf DAV-Sterbetafeln und Kommutationswerten
- Array-Formeln in Excel für Beitrags- und Verlaufsberechnungen
- Cache-Mechanismus in VBA für Performance-Optimierung
- 16 Dezimalstellen Rundungsgenauigkeit in allen Zwischenschritten

9.2 Nächste Schritte für Python-Migration

1. Import der Sterbetafeln als `pandas.DataFrame`
2. Implementation der Grundwerte-Funktionen (Modul `grundwerte.py`)
3. Implementation der Barwert-Funktionen (Modul `barwerte.py`)
4. Beitragsberechnung (Modul `tarif.py`)
5. Verlaufsberechnung (Funktion für alle Vertragsjahre)
6. Unit-Tests mit Excel-Vergleichswerten
7. Performance-Optimierung (Caching, Vektorisierung)
8. Dokumentation und Benutzeroberfläche

9.3 Kontaktinformationen

Für Rückfragen zur aktuariellen Logik oder Implementierungsdetails:

- Dokumentation erstellt: January 19, 2026
- Basis: Tarifrechner_KLV.xlsm
- Python-Version: 3.11+