# Arithmico Calc Anleitung (LaTeX-Version)

Stand 10.06.2023 Arithmico Calc Version 2.0.0 von Lennard Behrens

Autor: Ulrich Kalina

## Inhaltsverzeichnis

[Arithmico Calc Anleitung (LaTeX-Version) 1](#_Toc137313817)

[Inhaltsverzeichnis 1](#_Toc137313818)

[Einleitung 2](#_Toc137313819)

[Installation und Bedienoberfläche 2](#_Toc137313820)

[Installation 2](#_Toc137313821)

[Bedienoberfläche 2](#_Toc137313822)

[Kurztastenbefehle unter Windows 3](#_Toc137313823)

[Rechnerfunktionen an Beispielen 4](#_Toc137313824)

[1 Symbolische Konstanten 4](#_Toc137313825)

[2 Einfache Rechnungen und Standardfunktionen 5](#_Toc137313826)

[3 Benutzerdefinierte Variablen 9](#_Toc137313827)

[4 Benutzerdefinierte Funktionen 10](#_Toc137313828)

[5 Wertetabelle 12](#_Toc137313829)

[6 Gleichung mit einer Unbekannten lösen 13](#_Toc137313830)

[7 Lineares Gleichungssystem (LGS) lösen 15](#_Toc137313831)

[8 Ableitung einer Funktion an einer festen Stelle 15](#_Toc137313832)

[9 Bestimmtes Integral 16](#_Toc137313833)

[10 Vektoren 17](#_Toc137313834)

[11 Matrizen 18](#_Toc137313835)

[12 Stochastische Funktionen und Verteilungen 19](#_Toc137313836)

[13 Funktionsgrafen Plotter 21](#_Toc137313837)

[Index 22](#_Toc137313838)

## Einleitung

Arithmico Calc ist eine barrierefreie Software mit dem Funktionsumfang eines wissenschaftlichen Taschenrechners (WTR).

Mit dem Arithmico können mathematische Ausdrücke rein numerisch berechnet werden. Ab der Vollversion 2.0.0 besitzt er eine plot-Funktion zur Darstellung von Funktionsgrafen, aber keine weitergehende Grafikfunktionalität (GTR). Er führt auch keine algebraischen Umformungen durch (CAS) und ist nicht vom Anwender programmierbar. Die „erweiterte WTR-Funktionalität“ des Arithmico (z.B. Berechnung der Funktionsableitung an einer festen Stelle, Berechnung bestimmter Integrale, ...) entspricht den derzeitigen Anforderungen des Hessischen Landesabiturs.

Der Arithmico ist eine WEB-Anwendung, die sowohl online als auch offline (z.B. in Prüfungen) genutzt werden kann. Sie ist in den gängigen Internet-Browsern und damit auf einer großen Vielfalt an Endgeräten (PC, Laptop, Tablet, Smartphone, ...) unter verschiedenen Betriebssystemen nutzbar. Sie wird als Open Source Projekt entwickelt und ist im Rahmen der MIT-Lizenz frei verfügbar.

Die Bedienoberfläche des Arithmico zielt auf größtmögliche Barrierefreiheit ab - insbesondere auch beim Einsatz assistiver Technologien wie Screenreader, Sprachausgabe und Braillezeile.

## Installation und Bedienoberfläche

### Installation

**Online-Betrieb:** Die Internetseite https://arithmico.com/ in einem Browser öffnen.

**Offline-Betrieb:** Über die Seite https://arithmico.com/about die Offline-Version herunterladen (Offline-Version > Download offline version here). Einen neuen Ordner (z.B. mit dem Namen „Arithmico“) an einem beliebigen Ort (Festplatte, USB-Stick, ...) anlegen und die zip-Datei dort entpacken. Anschließend die Datei index.html in einem Browser öffnen. Gegebenenfalls eine Verknüpfung zu dieser Datei auf dem Desktop anlegen.

### Bedienoberfläche

#### Menüleiste

Am oberen Fensterrand des Arithmico befindet sich eine Menüleiste mit den Menüpunkten „Rechner“, „Einstellungen“, „Hilfe“ und „Über“, die als Web-Links implementiert sind.

#### Menüpunkt „Rechner“

Wenn der Menüpunkt „Rechner“ aktiv ist, befinden sich unter der Menüleiste ein Eingabefeld und ein Ausgabefeld mit jeweils einer Schaltfläche zum Löschen dahinter. Am unteren Fensterrand befinden sich die Schaltflächen „Definitionen“, „Verlauf“ und „Alles zurücksetzen“.

Der Menüpunkt „Rechner“ ist direkt nach dem Start des Arithmico immer aktiv und der Windows-Fokus befindet sich im Eingabefeld.

In das Eingabefeld kann ein mathematischer Ausdruck geschrieben werden. Nach Abschluss der Eingabe mit der ENTER-Taste wird der Ausdruck berechnet und das Ergebnis im Ausgabefeld angezeigt. Bei erneuter Eingabe von ENTER wechselt der Fokus zurück zum Eingabefeld.

Die TAB-Taste bewegt den Fokus reihum durch die Liste aller Bedienelemente (Ein- und Ausgabefeld, Menüpunkte, Schaltflächen), UMSCHALT+TAB tut dies in umgekehrter Reihenfolge. Die Inhalte der Textfelder werden dabei jeweils markiert.



#### Menüpunkt „Einstellungen“

Hier können Einstellungen zum Erscheinungsbild (Sprache, Schrift, farbliche Gestaltung, ...) und zum Rechnerverhalten (Anzahl signifikanter Stellen, landesspezifisches Zahlenformat bei Dezimalzahlen, Winkelmaßeinheit) vorgenommen werden. Beim Zahlenformat Deutsch wird das Komma (,) als Dezimalkomma und das Semikolon (;) als Trennzeichen zwischen Funktionsparametern verwendet.

#### Menüpunkt „Hilfe“

Hier werden alle Kurztastenbefehle des Arithmico sowie alle vordefinierten Funktionen (z.B. sin, log, exp, sqrt, ...) in Kurzform aufgelistet und erläutert. Bei Navigation durch die Einträge wird immer zunächst der Kurzbefehl und anschließend die Hilfe vorgelesen.

#### Menüpunkt „Über“

Dieser Menüpunkt enthält Informationen über die aktuell verwendete Arithmico-Version, sowie über das Arithmico-Projekt allgemein.

### Kurztastenbefehle unter Windows

ALT + i Eingabefeld fokussieren

ALT + o Ausgabefeld fokussieren

ALT + p Ein- und Ausgabefeld in Zwischenablage kopieren

STRG + i Eingabefeld löschen

STRG + o Ausgabefeld löschen

STRG + ALT + m Benutzerdefinierte Definitionen (Variablen und Funktionen) löschen

STRG + ALT + a Alles löschen

F5 Seite neu laden. Ein- und Ausgabe, Speicher und Protokoll werden gelöscht. Eine Bestätigung ist nicht erforderlich. Unverändert bleiben die auf der Seite Einstellungen festgelegten Parameter, wie Sprache, Dezimalstellen usw. Diese werden in den lokalen Website-Daten gespeichert und können dort gelöscht werden.

STRG + a Inhalt eines fokussierten Ein- bzw. Ausgabefeldes markieren

STRG + c Markierten Text in die Zwischenablage kopieren

STRG + v Zwischenablage in fokussiertes Eingabefeld einfügen

## Rechnerfunktionen an Beispielen

Eine vollständige Liste aller verfügbaren Standardfunktionen findet man unter dem Menüpunkt „Hilfe“.

Für die folgenden Beispiele gilt:

 • Die Eingaben beziehen sich auf ein leeres, fokussiertes Eingabefeld mit Schreibmarke (STRG + i).

 • Alle Eingaben müssen mit ENTER abgeschlossen werden.

 • Bei Ein- und Ausgaben wird von folgenden Einstellungen ausgegangen:
Signifikante Stellenzahl: 5
Sprache: DE
Zahlenformat: DE (Dezimalkomma, Trennzeichen für Funktionsparameter ist das Semikolon (;) ).

### 1 Symbolische Konstanten

#### 1.1 Kreiszahl \pi

Für die Kreiszahl \pi = 3,14159... kann die Buchstabenfolge pi als Konstantenbezeichner verwendet werden.

Beispiel 1.1.1: \pi soll näherungsweise als Dezimalzahl ausgegeben werden

Eingabe: pi

Ausgabe: 3,14159

Beispiel 1.1.2: 2 \pi = ?

Eingabe: 2\*pi

Ausgabe: 6,28319

#### 1.2 Euler'sche Zahl e

Für die Euler'sche Zahl e=2,71828... kann der Buchstabe e als Konstantenbezeichner verwendet werden.

Beispiel 1.2.1: e näherungsweise als Dezimalzahl ausgeben

Eingabe: e

Ausgabe: 2,71828

Beispiel 1.2.2: e^1,38629 =?

Eingabe: e^1,38629

Ausgabe: 3,99998

### 2 Einfache Rechnungen und Standardfunktionen

#### 2.1 KlaPoPuStri-Regel

Der Arithmico beachtet die Vorfahrtsregel "Klammer vor Potenz vor Punkt vor Strich" (KlaPoPuStri-Regel), Operatoren der gleichen Hierarchiestufe werden von links nach rechts bearbeitet. Als mathematische Klammern werden ausschließlich runde Klammern ( ) akzeptiert.

Beispiel 2.1.1: 2+3\*(4+5)^2 = ?

Eingabe: 2+3\*(4+5)^2

Ausgabe: 245

Beispiel 2.1.2: (4,31+8,3)/(3,26-2,5) = ?

Eingabe: (4,31+8,3)/(3,26-2,5)

Ausgabe: 16,59211

#### 2.2 Quadratwurzel von x

**sqrt(x)** Quadratwurzel von x.
Dabei ist x ist eine nicht-negative Zahl.

Beispiel 2.2.1: Quadratwurzel aus 25

Eingabe: sqrt(25)

Ausgabe: 5

#### 2.3 n-te Wurzel von x

**root(x; n)** n-te Wurzel von x.
Dabei sind der Radikand x und der Wurzelexponent n nicht-negative Zahlen.

Beispiel 2.3.1: Dritte Wurzel aus 8

Eingabe: root(8;3)

Ausgabe: 2

Beispiel 2.3.2: 1,5-te Wurzel aus 0,008

Eingabe: root(0,008;1,5)

Ausgabe: 0,04

#### 2.4 Logarithmus von x zur Basis b

**log(x; b)** Logarithmus von x zur Basis b.
Dabei sind x und b positive Zahlen.

Beispiel 2.4.1: log\_10(1000) Logarithmus von 1000 zur Basis 10

Eingabe: log(1000;10)

Ausgabe: 3

Beispiel 2.4.2: log\_2(32) Logarithmus von 32 zur Basis 2

Eingabe: log(32;2)

Ausgabe: 5

Beispiel 2.4.3: log\_0,8(1,25) Logarithmus von 1,25 zur Basis 0,8

Eingabe: log(1,25;0,8)

Ausgabe -1

#### 2.5 Dekadischer und natürlicher Logarithmus

**lg(x)** Logarithmus von x zur Basis 10 (Zehner-Logarithmus). Dabei ist x eine positive Zahl.

**ln(x)** Logarithmus von x zur Basis e (natürlicher Logarithmus). Dabei ist x eine positive Zahl.

Beispiel 2.5.1: lg(1000) =?

Eingabe: lg(1000)

Ausgabe: 3

Beispiel 2.5.2: ln(4) =? (siehe Beispiel 1.2.2)

Eingabe: ln(4)

Ausgabe: 1,38629

Beispiel 2.5.3: ln(e^2)

Eingabe: ln(e^2)

Ausgabe: 2

#### 2.6 Sinus von Winkel im Gradmaß

**sin(x)** Sinus von x

In den Einstellungen als Winkelmaß „Gradmaß“ wählen.

Beispiel 2.6.1: Sinus von 30° (degree)

Eingabe: sin(30)

Ausgabe: 0,5

Beispiel 2.6.2: Sinus von -270° (degree)

Eingabe: sin(-270)

Ausgabe: 1

#### 2.7 Tangens von Winkel im Gradmaß

**tan(x)** Tangens von x

In den Einstellungen als Winkelmaß „Gradmaß“ wählen.

Beispiel 2.7.1: Tangens von 135° (degree)

Eingabe: tan(135)

Ausgabe: -1

#### 2.8 Arkussinus im Gradmaß ausgeben

**asin(x)** Arkussinus von x ausgeben. Dabei ist x eine Zahl aus dem Intervall [-1; 1].

In den Einstellungen als Winkelmaß „Gradmaß“ wählen.

Beispiel 2.8.1: Arkussinus von 0,5 im Gradmaß (degree)

Eingabe: asin(0,5)

Ausgabe: 30

Beispiel 2.8.2: Arkussinus von 1 im Gradmaß (degree)

Eingabe: asin(1)

Ausgabe: 90

#### 2.9 Kosinus von einem Wert im Bogenmaß

**cos(x)** Kosinus von x

In den Einstellungen als Winkelmaß „Bogenmaß“ wählen.

Beispiel 2.9.1: Kosinus von 4/3 \pi

Eingabe: cos(4/3\*pi)

Ausgabe: -0,5

#### 2.10 Arkuskosinus Wert im Bogenmaß ausgeben

**acos(x)** Arkuskosinus von x. Dabei ist x eine Zahl aus dem Interval [-1; 1]

In den Einstellungen als Winkelmaß „Bogenmaß“ wählen.

Beispiel 2.10.1: Arkuskosinus von -0,5 im Bogenmaß

Eingabe: acos(-0,5)

Ausgabe: 2,0944

#### 2.11 Hyperbelsinus und Hyperbelkosinus

**sinh(x)** Sinus hyperbolicus von x

**cosh(x)** Kosinus hyperbolicus von x

Beispiel 2.11.1: sinh(1) + cosh(1) =?

Eingabe: sinh(1) + cosh(1)

Ausgabe: 2,71828

#### 2.12 n! - Fakultät einer Zahl n

**fact(n)** Fakultät von n, also n! = 1 \* 2 \* 3 \* ... \* n
Dabei ist n \in \N\_0 .

Beispiel 2.12.1: 6! =?

Eingabe: fact(6)

Ausgabe: 720

#### 2.13 |x| - Betrag einer Zahl x

**abs(x)** Betrag von x, also
|x| = x für x >= 0 und |x| = -x für x < 0.

Beispiel 2.13.1: |-3,215| =?

Eingabe: abs(-3,215)

Ausgabe: 3,215

#### 2.14 Maximum bzw. Minimum einer Werteliste

**max(x1; x2; x3; ...)** Maximum der Werte x1, x2, x3, ...

**min(x1; x2; x3; ...)** Minimum der Werte x1, x2, x3, ...

Beispiel 2.14.1: Gesucht sind größter und kleinster Wert in der Liste 2,745; -0,3155; 2,746; -1,84; 0,0002

Eingabe: max(2,745; -0,3155; 2,746; -1,84; 0,0002)

Ausgabe: 2,746

Eingabe: min(2,745; -0,3155; 2,746; -1,84; 0,0002)

Ausgabe: -1,84

#### 2.15 Einen Bruch kürzen

**fraction(x)** Kürze den Bruch x

Beispiel 2.15.1: Kürze den Bruch 3/6

Eingabe: fraction(3/6)

Ausgabe: 1 / 2

### 3 Benutzerdefinierte Variablen

Im Arithmico können benutzereigene Variablen für Zahlenwerte und Funktionen definiert und anschließend verwendet werden. Dafür gelten folgende Regeln:

 1. Die Bezeichner für benutzereigene Variablen oder Funktionen beginnen mit einem Buchstaben. Es können weitere Buchstaben, Ziffern oder Unterstriche (\_) folgen (ohne Leerzeichen dazwischen).

 2. Die Groß- und Kleinschreibung ist relevant, d.h. a (klein geschrieben) ist nicht gleich A (groß geschrieben).

 3. Die Bezeichner für vordefinierte Konstanten (\pi, e, ...) sollten nicht für selbstdefinierte Bezeichner verwendet werden, da die Konstantenbezeichner sonst überschrieben werden. Dies gilt auch für Funktionen, weil diese dadurch rekursiv definiert werden.

 4. Der Multiplikationsoperator muss immer verwendet werden - auch zwischen zwei Variablen (a\*b statt ab) oder zwischen einem Zahlenwert und einer Variablen (2\*x statt 2x).

 5. Bevor eine selbstdefinierte Variable in einem Rechenausdruck verwendet werden kann, muss ihr zuvor mit dem Zuweisungsoperator := ein konstanter Zahlenwert zugewiesen worden sein. Eine Variable behält ihren Wert so lange, bis ihr mit := ein neuer Wert zugewiesen wird.

 6. Den aktuellen Wert einer Variablen erhält man, indem man ihren Namen ins Eingabefeld schreibt. Eine Liste aller definierten Variablen- und Funktionsbezeichner ist über die Schaltfläche „Zeige Definitionen“ verfügbar. Es ist nicht möglich, einzelne Variable zu löschen.

#### 3.1 Variable definieren und verwenden

Beispiel 3.1.1 Der Variablen a wird der Wert 3 und der Variablen b der Wert 1/2 zugewiesen. Anschließend wird mit beiden gerechnet.

Eingabe: a:=3

Ausgabe: 3

Eingabe: b:=1/2

Ausgabe: 0,5

Eingabe: 2\*a - a\*b

Ausgabe: 4,5

#### 3.2 Liste aller aktuell verwendeten Variablen

Eine Liste aller aktuell verwendeten Variablen und Funktionen erhält man über die Schaltfläche „Definitionen“.

#### 3.3 Wert einer Variablen überschreiben

Beispiel 3.3.1 Nachdem die Variablen a und b wie in 3.1 definiert wurden, soll a nun den Wert 5 erhalten. Anschließend wird wieder mit beiden gerechnet.

Eingabe: a:=5

Ausgabe: 5

Eingabe: 2\*a - a\*b

Ausgabe: 7,5

#### 3.4 Alle Variablen löschen

Mit der Schaltfläche „Definitionen“ und dann „Zurücksetzen“ werden alle benutzerdefinierten Variablen und Funktionsdefinitionen gelöscht.

### 4 Benutzerdefinierte Funktionen

Für die Bezeichner benutzerdefinierter Funktionen gelten die gleichen Schreibregeln wie für Variablen. Zusätzlich gilt hier:

 1. An den Namen des Funktionsbezeichners wird in runden Klammern eine Liste mit (formalen) Parametern angehängt in der Form f(x; y; z). Die Liste kann auch nur ein Element enthalten, also die Form f(x) besitzen. Zwischen dem Funktionsnamen und der öffnenden Klammer darf kein Leerzeichen stehen.

 2. Mit dem Zuweisungsoperator := wird anschließend ein Ausdruck (Funktionsterm) mit den Parametern x, y, ... zugewiesen.

 3. Danach kann der Funktionsbezeichner gefolgt von runden Klammern mit konkreten Zahlen als Funktionsargumenten verwendet werden. Die Anzahl der Argumente muss mit der Anzahl der Parameter übereinstimmen.

 4. Funktionen können auch ohne Angabe eines Funktionsbezeichners mit dem Operator -> definiert werden.
Beispiel Quadratfunktion: x -> x^2 (siehe auch 5., 8.1 bzw. 9.1).

#### 4.1 Funktion definieren und verwenden

Beispiel 4.1.1 Die Funktion f(x) = x^2 -2x wird definiert. Anschließend werden die Funktionswerte für verschiedene x-Werte ausgegeben.

Eingabe: f(x):=x^2 -2\*x

Ausgabe: (x: any) → x^2 - 2 \* x

Eingabe: f(1)

Ausgabe: -1

Eingabe: f(2)

Ausgabe: 0

Eingabe: f(2,5)

Ausgabe: 1,25

Beispiel 4.1.2: Die Funktionswerte der Polynomfunktion
f(x)= x^3 -4x^2 +3 sollen nacheinander an den Stellen x=-1, x=0, x=1 und x=2 berechnet werden.

Eingabe: f(x):=x^3 -4\*x^2 +3

Ausgabe: (x: any) → x^3 - 4 \* x^2 + 3

Eingabe: f(-1)

Ausgabe: -2

Eingabe: f(0)

Ausgabe: 3

Eingabe: f(1)

Ausgabe: 0

Eingabe: f(2)

Ausgabe: -5

Beispiel 4.1.3: Die Funktion A(x,y) = x\*y wird definiert, um den Flächeninhalt eines Rechtecks mit den Seitenlängen x und y zu berechnen. Anschließend wird die Funktion mit verschiedenen Argumenten aufgerufen.

Eingabe: A(x;y):=x\*y

Ausgabe: (x: any; y: any) → x \* y

Eingabe: A(4;5)

Ausgabe: 20

Eingabe: t:=1/2

Ausgabe: 0,5

Eingabe: A(3;t)

Ausgabe: 1,5

#### 4.2 Funktionswerte einer Sinusfunktion

In den Einstellungen als Winkelmaß „Bogenmaß“ wählen.

Beispiel 4.2.1 Von der Funktion f(t) =2 sin(t-1/2 \pi) sollen nacheinander die Funktionswerte f(0), f(1/3 \pi) und f(1/2 \pi) ausgegeben werden.

Eingabe: f(t):=2\*sin(t-1/2\*pi)

Ausgabe: (t: any) → 2 \* sin(t - 1 / 2 \* pi)

Eingabe: f(0)

Ausgabe: -2

Eingabe: f(1/3\*pi)

Ausgabe: -1

Eingabe: f(1/2\*pi)

Ausgabe: 0

#### 4.3 Funktionswerte einer Exponentialfunktion

Beispiel 4.3.1 Berechne nacheinander für die Funktion K(n)=1000\*1,05^n die Funktionswerte K(0), K(1) und K(2)

Eingabe: K(n):=1000\*1,05^n

Ausgabe: (n: any) → 1000 \* 1,05^n

Eingabe: K(0)

Ausgabe: 1000

Eingabe: K(1)

Ausgabe: 1050

Eingabe: K(2)

Ausgabe: 1102,5

### 5 Wertetabelle

**table(f, u?, v?, w?)** Gibt zu der Funktion f eine Wertetabelle mit Wertepaaren [x; f(x)] für x-Werte aus dem Intervall [u; v] aus. Dabei gibt w die Schrittweite der x-Werte an.

Beispiel 5.1 Gib für die Funktion f(x)=x^2 eine Wertetabelle für die x-Werte aus dem Intervall [-2; 3] mit der Schrittweite 1 aus.

Eingabe: f(x):=x^2

Ausgabe: (x: any) → x^2

Eingabe: table(f; -2; 3; 1)

oder kurz:

Eingabe: table(x->x^2; -2; 3; 1)

Ausgabe: [[-2; 4]; [-1; 1]; [0; 0]; [1; 1]; [2; 4]; [3; 9]]

### 6 Gleichung mit einer Unbekannten lösen

**nsolve(u; v?; w?)** Bestimmt alle Lösungen der Gleichung u im Intervall [v; w]. Werden die Untergrenze v und die Obergrenze w nicht angegeben, werden die Lösungen im Intervall [-20; 20] bestimmt.
Der Variablen, nach der die Gleichung u aufgelöst werden soll, darf vorher noch nicht mit := ein Wert zugewiesen worden sein. Gegebenenfalls Definitionen löschen.

#### 6.1 Lineare Gleichung ohne Intervallangabe lösen

Beispiel 6.1.1 Berechne die Lösungen der Gleichung
3x -5 = x +7 im Intervall [-20; 20].

Eingabe: nsolve(3\*x -5 = x +7)

Ausgabe: [6]

#### 6.2 Quadratischen Gleichung ohne Intervallangabe

Beispiel 6.2.1 Berechne die Lösungen der Gleichung
2x^2 +4x -30 =0 im Intervall [-20; 20].

Eingabe: nsolve(2\*x^2 +4\*x -30 =0)

Ausgabe: [-5; 3]
Es gibt die Lösungen: x\_1 = -5 und x\_2 =3

Beispiel 6.2.2 Berechne die Lösungen der Gleichung
x^2 -x +0,25 =0 im Intervall [-20; 20].

Eingabe: nsolve(x^2 -x +0,25 =0)

Ausgabe: [0,5] Es gibt nur eine Lösung: x = 0,5

Beispiel 6.2.3 Berechne die Lösungen der Gleichung
x^2 =-4 im Intervall [-20; 20].

Eingabe: nsolve(x^2 =-4)

Ausgabe: [] Es gibt keine Lösung (leere Lösungsmenge).

#### 6.3 Kubische Gleichung ohne Intervallangabe

Beispiel 6.3.1 Berechne die Lösungen der Gleichung
x^3 +x^2 -17x +15 =0 im Intervall [-20; 20].

Eingabe: nsolve(x^3 +x^2 -17\*x +15 =0)

Ausgabe: [-5; 1; 3]

#### 6.4 Lösen einer Gleichung mit Intervallangabe

Beispiel 6.4.1 Berechne die Lösungen der Gleichung x -1 = 20 mit nsolve zunächst **ohne** explizite Intervallangabe.

Eingabe: nsolve(x-1 = 20)

Ausgabe: []
Die Lösung x=21 liegt außerhalb des Standardintervalls [-20; 20]. Daher wird hier die leere Menge als Lösungsmenge angezeigt.

Beispiel 6.4.2 Berechne die Lösungen der Gleichung
x -1=20 im Intervall [0; 30].

Eingabe: nsolve(x -1 =20; 0; 30)

Ausgabe: [21]
Die Lösung x=21 liegt jetzt innerhalb des angegebenen Intervalls [0; 30]

#### 6.5 Nullstellen der Sinusfunktion im Intervall [0; 7]

Beispiel 6.5.1 Berechne die Nullstellen der Standard-Sinus-Funktion sin(x) im Intervall [0; 7].

Eingabe: nsolve(sin(x)=0; 0; 7)

Ausgabe: [0; 3,14159; 6,28319] Nullstellen 0, \pi und 2\pi (bei Einstellung Winkelmaß=Bogenmaß)

#### 6.6 Nullstellen einer Exponentialfunktion

Beispiel 6.6.1 Berechne die Nullstellen der Funktion
f(x) = (x^2 -2x)\*e^{0,5 x} im Intervall
[-20; 20].

Eingabe: nsolve((x^2 -2\*x)\*e^(0,5\*x)=0)

Ausgabe: [0; 2]

#### 6.7 Lösungen einer Exponentialgleichung

Beispiel 6.7.1 Berechne die Lösungen der Gleichung
10 e^{0,1 t} = 50 -40 e^{-0,1 t}
im Intervall [-20; 20].
(Variable t darf noch nicht definiert worden sein!)

Eingabe: nsolve(10\*e^(0,1\*t) = 50-40\*e^(-0,1\*t))

Ausgabe: [0; 13,862944]

#### 6.8 Nullstellen einer rationalen Funktion

Beispiel 6.8.1 Berechne die Nullstellen der Funktion
f(x) = (x+2)^2 / (x-1) im Intervall [-20; 20].

Eingabe: nsolve((x+2)^2/(x-1)=0)

Ausgabe: [-2]

### 7 Lineares Gleichungssystem (LGS) lösen

**lsolve(u; v; ...)** Löst ein System von linearen Gleichungen mit den Gleichungen u, v, ... Das LGS muss eindeutig lösbar sein. Die Anzahl der Gleichungen muss mit der Anzahl der Unbekannten übereinstimmen.

#### 7.1 LGS mit zwei Gleichungen und zwei Unbekannten

Beispiel 7.1.1 Berechne die Lösungen des LGS
I: x +y =5
II: 2x -y =1

Eingabe: lsolve(x +y =5; 2\*x -y =1)

Ausgabe: [x = 2; y = 3]

#### 7.2 LGS mit drei Gleichungen und drei Unbekannten

Beispiel 7.2.1 Berechne die Lösungen des LGS
I: z +y +x =4
II: 2x +2z =y-1
III: x = z+3

Eingabe: lsolve(z +y +x =4; 2\*x +2\*z =y-1; x = z+3)

Ausgabe: [z = -1; y = 3; x = 2]

### 8 Ableitung einer Funktion an einer festen Stelle

**nderive(f(x); x; g?)** Ableitung der Funktion f an der Stelle x. Wird der Grad g nicht angegeben, wird die erste Ableitung berechnet.

#### 8.1 Ableitung einer Polynomfunktion

Beispiel 8.1.1 Gegeben ist die Funktion f(x) = x^2. Berechne die 1. Ableitung an der Stelle 3, also f '(3).

Eingabe: f(x) := x^2

Ausgabe: (x: any) → x^2

Eingabe: nderive(f ; 3)

Ausgabe: 6

oder kürzer:

Eingabe: nderive((x)->(x^2); 3)

Ausgabe: 6

#### 8.2 Ableitung der e-Funktion

Beispiel 8.2.1 Gegeben ist die Funktion f(x) = e^x. Berechne die 1. Ableitung an der Stelle 1, also f '(1)

Eingabe: nderive((x)->e^x; 1)

Ausgabe: 2,71828

#### 8.3 Ableitung der natürlichen Logarithmusfunktion

Beispiel 8.3.1 Berechne die 1. Ableitung der natürlichen Logarithmusfunktion f(x) = ln(x) an der Stelle 4, also f '(4)

Eingabe: nderive((x)->ln(x); 4)

Ausgabe: 0,25

#### 8.4 Ableitung der Sinus-Funktion

Beispiel 8.4.1 Berechne die 1. Ableitung der Sinusfunktion
f(x) = sin(x) an der Stelle \pi /4 , also f '(\pi /4)

Eingabe: nderive((x)->sin(x); pi /4)

Ausgabe: 0,70711

### 9 Bestimmtes Integral

**nintegrate(f(x); u; v)** Berechne das bestimmte Integral der Funktion f mit Untergrenze u und Obergrenze v.

#### 9.1 Bestimmtes Intral über einer Polynomfunktion

Beispiel 9.1.1 Berechne dnas bestimmte Integral
\int\_0^1 x^2 dx .

Eingabe: nintegrate((x)→x^2; 0; 1)

Ausgabe: 0,33333

#### 9.2 Bestimmtes Integral über einer trigonometrischen Funktion

Beispiel 9.2.1 Berechne das bestimmte Integral
\int\_0^\pi sin(x) dx .

Eingabe: nintegrate((x) -> sin(x); 0; pi)

Ausgabe: 2

#### 9.3 Bestimmtes Integral über einer Exponentialfunktion

Beispiel 9.3.1 Berechne das bestimmte Integral
\int\_0^1 e^x dx .

Eingabe: nintegrate((x) -> e^x; 0; 1)

Ausgabe: 1,71828

Beispiel 9.3.2 \int\_0^\infty e^(-x) dx soll näherungsweise berechnet werden. Dazu wird die obere Integralgrenze mit b = 20 so gewählt, dass
der Funktionswert e^(-b) = e^(-20)
= 2,06115 \* 10^(-9) dicht bei 0 liegt.

Eingabe: nintegrate((x)->e^(-x);0;20)

Ausgabe: 1

### 10 Vektoren

**[a1; a2; ...]** Vektoren werden als Liste von Koordinaten in eckigen Klammern geschrieben.

#### 10.1 Vektoraddition

**[a1; a2; a3] + [b1; b2; b3]**
Berechnet die Vektorsumme von \vec{a} =(a1; a2; a3) und \vec{b} =(b1; b2; b3).

Beispiel 10.1.1 Berechne die Summe der Vektoren
(1; -2; 2) + (2; 1; 0)

Eingabe: [1; -2; 2] + [2; 1; 0]

Ausgabe: [3; -1; 2]

#### 10.2 Skalarprodukt

**[a1; a2; a3] \* [b1; b2; b3]**Berechnet das Skalarprodukt von \vec{a} =(a1; a2; a3) und \vec{b} =(b1; b2; b3).

Beispiel 10.2.1 Berechne das Skalarprodukt der Vektoren
(1; -2; 2) \* (2; 1; 0)

Eingabe: [1; -2; 2] \* [2; 1; 0]

Ausgabe: 0

#### 10.3 Vektorprodukt

**cross([a1; a2; a3]; [b1; b2; b3])**
Berechnet das Vektorprodukt von \vec{a} =(a1; a2; a3) und \vec{b} =(b1; b2; b3).

Beispiel 10.3.1 Berechne das Vektorprodukt der Vektoren
(1; -2; 2) \times (2; 1; 0)

Eingabe: cross([1; -2; 2] ; [2; 1; 0])

Ausgabe: [-2; 4; 5]

#### 10.4 Vektorlänge

**abs([a1;a2;a3])** Berechnet die Länge des Vektors (a1;a2;a3).

Beispiel 10.4.1 Berechne die Länge (den Betrag) des Vektors (1; -2; 2)

Eingabe: length([1; -2; 2])

Ausgabe: 3

### 11 Matrizen

**[[a1;b1;...] ; [a2;b2;...] ; ...]** Matrizen werden als Liste von (Zeilen-) Vektoren in eckigen Klammern geschrieben.

#### 11.1 Matrizensumme

**A + B** Berechnet die Matrizensumme der Matrizen A und B

Beispiel 11.1.1 Berechne die Matrizensumme
\mat{1 &2 \\ 3&4} + \mat{5&6 \\ 7&8}

Eingabe: [[1; 2]; [3; 4]]+[[5; 6]; [7; 8]]

Ausgabe: [[6; 8]; [10; 12]]

#### 11.2 Matrizenprodukt

**A \* B** Berechnet das Produkt der Matrizen A und B

Beispiel 11.2.1 Berechne das Matrizenprodukt
\mat{1 &2 \\ 3&4} \* \mat{5&6 \\ 7&8}

Eingabe: [[1; 2]; [3; 4]]\*[[5; 6]; [7; 8]]

Ausgabe: [[19; 22]; [43; 50]]

Beispiel 11.2.2 Berechne das Matrizenprodukt \mat{1 &2 \\ 3&4} \* \mat{-2 &1 \\ 1,5 & -0,5}

Eingabe: [[1; 2]; [3; 4]]\*[[-2; 1]; [1,5; -0,5]]

Ausgabe: [[1; 0]; [0; 1]]

#### 11.3 Inverse Matrix

**matrix:inverse(A)** Berechnet die inverse Matrix A^{-1} zu einer (quadratischen) Matrix A.

Beispiel 11.3.1 Berechne die inverse Matrix \mat{1 &2 \\ 3&4}

Eingabe: matrix:inverse([[1; 2]; [3; 4]])

Ausgabe: [[-2; 1] ; [3/2 ; -1/2]]

### 12 Stochastische Funktionen und Verteilungen

#### 12.1 Binomialkoeffizient {n \choose k}

**binco(n; k)** Berechnet den Binomialkoeffizient {n \choose k} „n über k“

Beispiel 12.1.1 Berechne den Binomialkoeffizient
{6 \choose 2}.

Eingabe: binco(6; 2)

Ausgabe: 15

#### 12.2 Binomialverteilung B(n; p; k)

**binom(n; p; k)** Binomialverteilung mit den Parametern
n = Anzahl der Stufen des mehrstufigen Zufallsversuchs, p =Trefferwahrscheinlichkeit,
k = Trefferanzahl

Beispiel 12.2.1 Berechne B(6; 0,5; 2) = P(X=2) mit n=6; p=0,5 und k=2, X binomialverteilt.

Eingabe: binom(6; 0,5; 2)

Ausgabe: 0,23438

#### 12.3 Kumulierte Binomialverteilung F(n; p; k)

**cbinom(n; p; k)** Kumulierte Binomialverteilung mit
n = Anzahl der Stufen des mehrstufigen Zufallsversuchs, p =Trefferwahrscheinlichkeit, k = Trefferanzahl

Beispiel 12.3.1 Berechne F(1000; 0,45; 421) = P(X<=421) mit n=1000, p=0,45 und k=421, X binomialverteilt

Eingabe: cbinom(1000; 0,45; 421)

Ausgabe: 0,03481

#### 12.4 Kumulierte Binomialverteilung 1 - F(n; p; k)

Beispiel 12.4.1 Berechne 1 - F(1000; 0,45; 421) = P(X > 421) mit n=1000 und p=0,45, k=421,
X binomialverteilt

Eingabe: 1 - cbinom(1000; 0,45; 421)

Ausgabe: 0,96519

#### 12.5 Gauß'sche Dichtefunktion \phi(x) (Standardnormalverteilung)

**normal(x)** Standardnormalverteilung mit Erwartungswert 0 und Varianz 1.

Beispiel 12.5.1 Berechne \phi(0) mithilfe der Definition der Dichtefunktion
\phi(x) = 1/ \sqrt{2\pi} \*e^{-0,5\*x^2}

Eingabe: phi(x):=1/ sqrt(2\*pi) \*e^(-0,5\*x^2)

Ausgabe: (x: any) → 1 / sqrt(2 \* pi) \* e^(-0,5 \* x^2)

Eingabe: phi(0)

Ausgabe: 0,39894

Bespiel 12.5.2 Berechne \phi(0) mithilfe des Befehls normal(x) (Standardnormalverteilung)

Eingabe: normal(0)

Ausgabe: 0,39894

#### 12.6 Gauß'sche Integralfunktion \Phi(z) (Kumulierte Standardnormalverteilung)

**cnormal(z)** Kumulierte Standardnormalverteilung mit Erwartungswert 0 und Varianz 1. Dabei ist \Phi(z) = \int\_{-\infty}^z \phi(t) dt und
\phi(t) = 1/ \sqrt{2\pi} \*e^{-0,5\*t^2}

Beispiel 12.6.1 Berechne \Phi(0,5)

Eingabe: cnormal(0,5)

Ausgabe: 0,69146

#### 12.7 Gauß'sche Fehlerfunktion erf(x)

**erf(x)** Gaußsche Fehlerfunktion

Beispiel 12.7.1 Berechne erf(0,45)

Eingabe: erf(0,45)

Ausgabe: 0,47548

#### 12.8 Arithmetischer Mittelwert

**avg(x)** Berechnet den arithmetischen Mittelwert einer Werteliste.

Beispiel 12.8.1 Berechne das arithmetische Mittel der Werte 1 ; 3 ; 5 ; 9 ; 12, also \overline{x} = (1 + 3 + 5 + 9 + 12)/5 = 30/5 = 6

Eingabe: avg(1;3;5;9;12)

Ausgabe: 6

#### 12.9 Varianz einer Stichprobe

**var(x)** Berechnet die Stichprobenvarianz \sigma^2 einer Werteliste.

Beispiel 12.9.1 Berechne die Varianz \sigma^2 der Liste 1 ; 3 ; 5 ; 9 ; 12, also \sigma ^2=((1-6)^2+(3-6)^2+(5-6)^2+(9-6)^2+(12-6)^2))/5=(25+9+1+9+36)/5=80/5=16

Eingabe: var(1;3;5;9;12)

Ausgabe: 16

#### 12.10 Standardabweichung

**sd(x)** Berechnet die Standardabweichung \sigma einer Werteliste.

Beispiel 12.10.1 Berechne \sigma für die Liste 1 ; 3 ; 5 ; 9 ; 12

Eingabe: sd(1;3;5;9;12)

Ausgabe: 4

### 13 Funktionsgrafen Plotter

**plot(f; g; ...; xMin?; xMax?; yMin?; yMax?)** Erzeugt einen Plot der Funktionen f, g usw. im optionalen Bereich xMin bis xMax auf der x-Achse und von yMin bis yMax auf der y-Achse. Der Standardwert für xMin ist -10. Der Standardwert für xMax ist 10.
Über das Download-Symbol kann die Grafik als PDF-Datei heruntergeladen werden, wobei die Beschriftung wahlweise in Schwarzschrift oder in Braille gewählt werden kann.

Beispiel 13.1 Zeichne die Funktionsgrafen der Funktionen f(x) = x^2 und
g(x) =2^x für x-Werte aus dem Intervall [-2; 3]

Eingabe: plot(x->x^2; x->2^x; -2; 3)

Ausgabe:

 

## Index

**Befehl /** Beschreibung Seite

acos(x)

Arkuskosinus von x 8

asin(x)

Arkussinus von x 7

binco(n; k)

Berechnet den Binomialkoeffizient {n \choose k} „n über k“ 19

binom(n; p; k)

Binomialverteilung mit den Parametern n, p und k 19

cbinom(n; p; k)

Kumulierte Binomialverteilung 20

cnormal(z)

Kumulierte Standardnormalverteilung (Gauß’sche Integralfunktion) 21

cos(x)

Kosinus von x 8

cosh(x)

Kosinus hyperbolicus von x 8

e

Euler'sche Zahl e=2,71828... 5

erf(x)

Gaußsche Fehlerfunktion 21

fact(n)

Fakultät von n 8

lg(x)

Logarithmus von x zur Basis 10 (Zehner-Logarithmus) 6

ln(x)

Logarithmus von x zur Basis e (natürlicher Logarithmus) 6

log(x; b)

Logarithmus von x zur Basis b 6

lsolve(u; v; ...)

Löst das lineare Gleichungssystem mit den Gleichungen u, v, ... 15

Matrix [[a1;b1;...] ; [a2;b2;...] ; ...]

Matrizen werden als Liste von (Zeilen-) Vektoren in eckigen Klammern geschrieben. 18

matrix:inverse(A)

Berechnet die inverse Matrix zu Matrix A 19

Matrizenprodukt A \* B

Berechnet dasProdukt der Matrizen A und B 19

Matrizensumme A + B

Berechnet die Summe der Matrizen A und B 18

max(x1; x2; x3; ...)

Maximum der Werte x1, x2, x3, ... 9

min(x1; x2; x3; ...)

Minimum der Werte x1, x2, x3, ... 9

nderive(f(x); x; g?)

Ableitung der Funktion f an der Stelle x 16

nintegrate(f(x); u; v)

Berechnet das bestimmte Integral der Funktion f in den Grenzen von u bis v 16

normal(x)

Standardnormalverteilung mit Erwartungswert 0 und Varianz 1 20

nsolve(u; v?; w?)

Bestimmt alle Lösungen der Gleichung u im Intervall [v; w] 13

pi

Kreiszahl \pi = 3,14159 4

plot(f; g; ...; xMin?; xMax?; yMin?; yMax?)

Erzeugt einen Plot der Funktionen f, g usw. 22

root(x; n)

n-te Wurzel von x 6

sd(x)

Berechnet die Standardabweichung \sigma einer Werteliste 21

sin(x)

Sinus von x 7

sinh(x)

Sinus hyperbolicus von x 8

Skalarprodukt [a1; a2; a3] \* [b1; b2; b3]

Berechnet das Skalarprodukt der Vektoren (a1; a2; a3) und (b1; b2; b3) 18

sqrt(x)

Quadratwurzel von x 5

table(f, u?, v?, w?)

Gibt zu der Funktion f eine Wertetabelle mit Wertepaaren [x; f(x)] für x-Werte aus dem Intervall [u; v] aus 13

tan(x)

Tangens von x 7

var(x)

Berechnet die Stichprobenvarianz \sigma^2 einer Werteliste 21

Vektorlänge abs([a1;a2;a3])

Berechnet die Länge des Vektors (a1;a2;a3) 18

Vektorprodukt cross([a1; a2; a3] ; [b1; b2; b3])

Berechnet das Vektorprodukt der Vektoren (a1; a2; a3) und (b1; b2; b3) 18

Vektorsumme [a1; a2; a3] + [b1; b2; b3]

Berechnet die Summe der Vektoren (a1; a2; a3) und (b1; b2; b3) 17