ETH zürich

Züchtungslehre I+II

Birgit Gredler and Peter von Rohr

Qualitas AG

Inhalt

Who Is Who

Dozierende

- Birgit Gredler
- Peter von Rohr

Studierende

- Studiengang
- Erfahrungen mit Tierzucht
- Motivation für diese Vorlesung

Ziele der Vorlesung

- Verstehen der Grundlagen
- Erklärung von Zusammenhängen
- Beurteilung von Aussagen (siehe folgende Zitate)

Zitate

Zitat 1: Leserbrief im Schweizer Bauer

"[...] Ich habe noch niemanden getroffen, der mir diese Zuchtwerte erklären kann. Eine Kuh von mir hat einen Zuchtwert von -900 und gibt immer noch Milch. [...] "

Zitat 2: swissherdbookbulletin 5/15

"Bei der Auswahl von Kühen für die Zuchtprogramme sollten also auch Eigenleistungen, Leistungen von Vorfahren und Blutlinien stimmen."

Zitat 3: Mein Praktikumsbauer

"Wozu brauche ich Zuchtwerte, mein Züchterauge sieht die guten Kühe auch so."

Administrative Angelegenheiten

Sprache

- Deutsch
- Fachbegriffe Englisch

Webseite

- Link: http://charlotte-ngs.github.io/LivestockBreedingAndGenomics/
- Syllabus, Folien und Übungen werden auf der Webseite verfügbar sein

Administrative Angelegenheiten II

Keine Vorlesung am 9. Oktober 2015

Mögliche Alternativen sind

- Zusatztermin
- Während dreier Wochen eine Lektion kompensieren
- Entscheidung vor 9. Oktober 2015

Kreditpunkte

Leistungsnachweis: schriftliche Prüfung am 18.12.2015

Vorlesungsprogramm

Vorlesungsprogramm auf der Webseite unter "Syllabus" verfügbar

	Datum	Thema	Wer
1	18.09	Einführung in die Vorlesung, R Selektionsindex	BG PvR
2	25.09	Selektionsindex mehrere Merkmale	PvR
3	02.10	Verwandtschaftsmatrix und ihre Inverse	PvR
4	09.10	keine Vorlesung	
5	16.10	Korrektur für fixe Effekte	PvR
6	23.10	Varianzanalyse	PvR
7	30.10	Varianzkomponentenschätzung	PvR
8	06.11	Varianzkomponentenschätzung Teil II BLUP ein Merkmal	PvR BG
9	13.11	BLUP mehrere Merkmale, wirtschaftliche Gewichte	BG

Vorlesungsprogramm II

10	20.11	Linkage disequilibrium	BG
11	27.11	Genomische Selektion	BG
12	04.12	Genomische Selektion	BG
13	11.12	Genom-weite Assoziationsstudien	BG
14	18.12	Prüfung	BG, PvR

Ablauf einer Vorlesung

Typ gemäss Vorlesungsverzeichnis

- Typ G, d.h. Vorlesung und Übung

Unser Anliegen

- Möglichst viel Interaktion, denn so lernen wir am meisten
- Möglichst wenig Konsumation, denn dabei lernen wir nichts

Ablauf

- Ab kommender Woche
- $3G = 1U + 2V$ wobei U: 9-10, V: 10-12
- Pausen

Voraussetzungen für diese Vorlesung

Streng genommen

- KEINE
- Grundlegende Begriffe und wichtige Konzepte werden erläutert

Hilfreich sind ...

- Grundbegriffe der quantitativen Genetik (Bachelor)
- Grundbegriffe der Statistik (Erwartungswert, Varianz, Schätzung)
- Elementare Kenntnisse der linearen Algebra (Vektoren, Matrizen)
- Erste Erfahrung mit Programmiersprache (Matlab) oder Statistiksoftware (R)

Übungen

- Zu jedem Vorlesungsblock wird es eine Übung geben
- Übungsstunde steht zur Bearbeitung der Aufgaben zur Verfügung
- Lösungsvorschläge zu einem späteren Zeitpunkt
- Stil der Übungsaufgaben: Bearbeitung einer Fragestellung mit R (oder anderer Programmiersprache)

Übungen II: Weshalb programmieren?

Vorteile der Problemlösung mit einem Programm

- Datenmengen verlangen nach Tools zur effizienten Bearbeitung und Analyse
- Flexibilität: Excel und co. sind zu wenig flexibel und zu beschränkt (z. Bsp. Anzahl Zeilen und Kolonnen)
- Reproduzierbarkeit und Nachvollziehbarkeit
- Repetivite Arbeiten werden von einem Computerprogramm besser erledigt
- besseres Verständnis eines Problems
- Programmierfähigkeiten können Sie immer wieder gebrauchen (Praktika, Masterarbeit, Doktorarbeit, Job)

Übungen III: Ihre Erfahrungen

Umfrage nach Programmiererfahrung

- Kennen Sie eine/mehrere Programmiersprachen, wenn ja welche?
- Wie erledigen Sie Datenverarbeitungsjobs? (Semesterarbeit, Praktika, Bachelorarbeit)
- Was hat Sie bis jetzt daran gehindert das Programmieren zu erlernen?
- In welchen Veranstaltungen (Vorlesungen, Übungen, Praktika) wurden Sie schon mit Programmiersprachen konfrontiert und was sind Ihre Erfahrungen

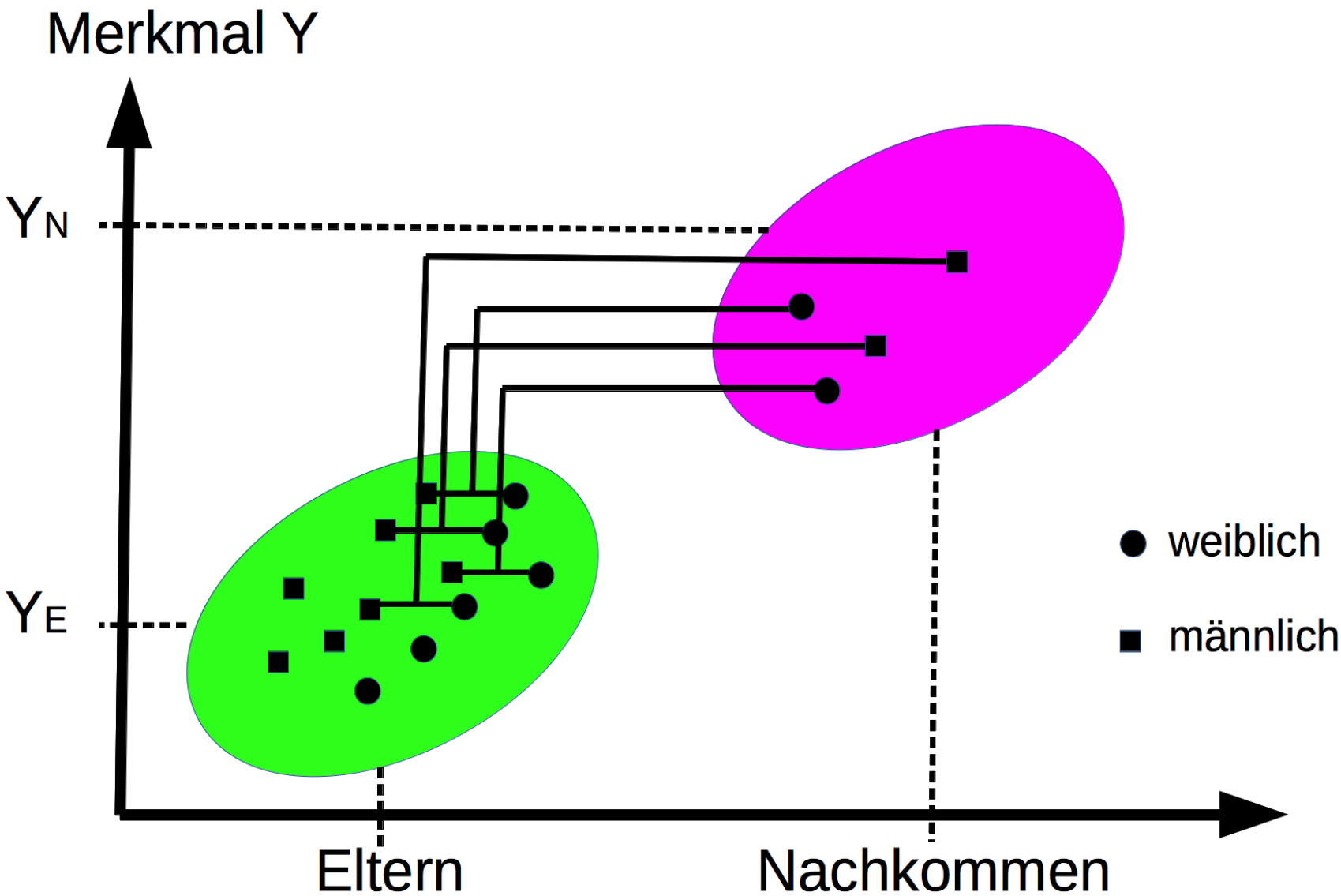
Einführung in die Theorie des Selektionsindexes

- Allgemeine Tierzucht - Ein kurzer Rückblick
- Zuchtziel - verschiedene Formulierungen
- Verschiedene Formen der Selektion
- Index-basierte Selektion

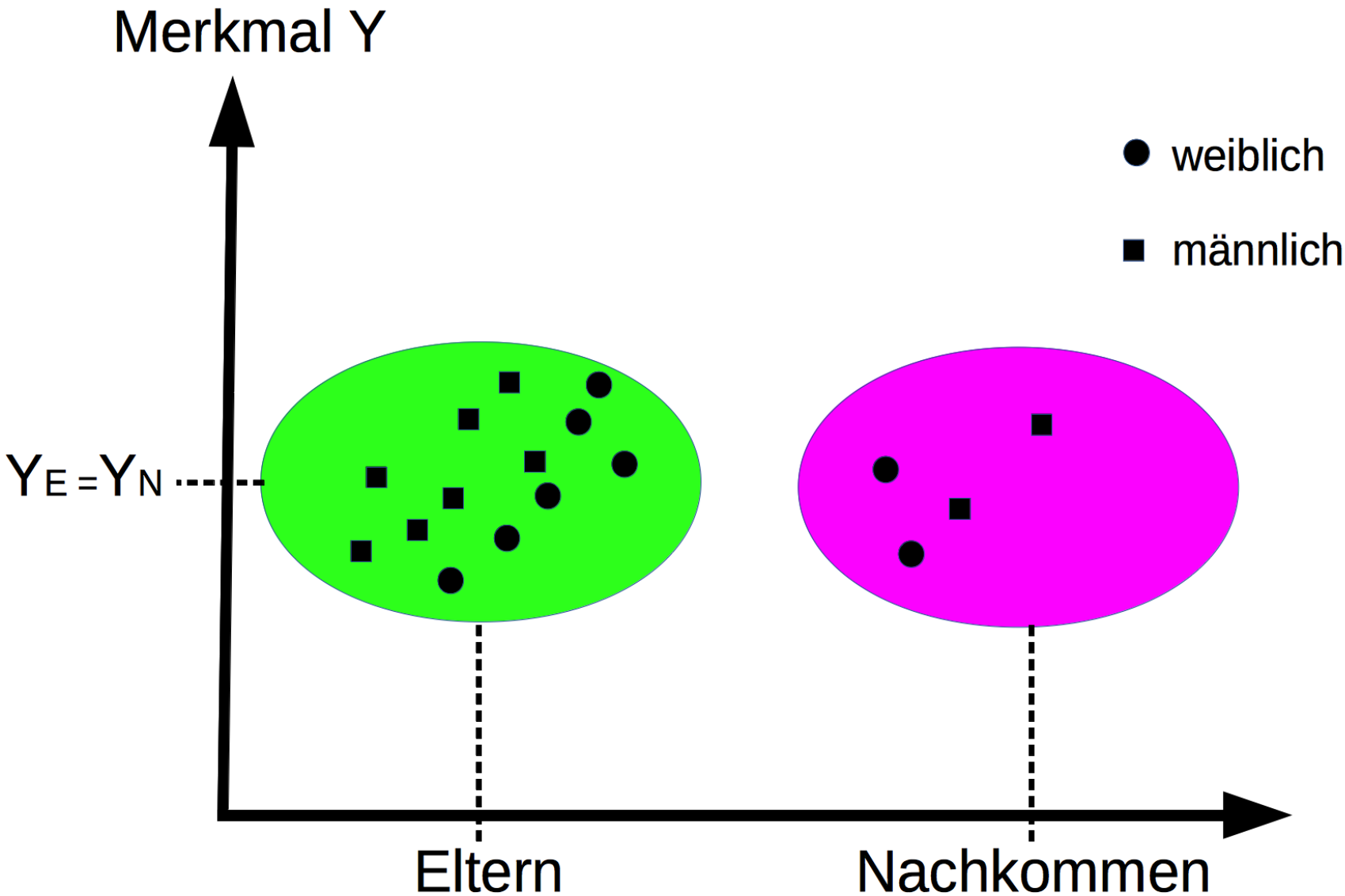
Tierzucht - Rückblick

- Voraussetzung: *Zuchtziel*, welches einer Idealvorstellung eines Tieres in verschiedenen Merkmalen entspricht
- *Züchten*: geziehlte Auswahl (Selektion) von Elterntieren mit der Absicht, dass die Nachkommen im Durchschnitt dem Zuchtziel besser entsprechen als der Durchschnitt der Elterngeneration
- Gegensatz zu Zucht steht die *Vermehrung* oder *Produktion*, welche keine gezielte Anpaarung von Elterntieren beinhaltet
- Umsetzung der Erreichung des Zuchtziels in einem *Zuchtprogramm*

Züchtung



Vermehrung



Arten der Selektion

	Natürliche Selektion NS	Künstliche Selektion KS
Überleben	Umwelt bestimmt, welche Tiere in Population überleben	Mensch bestimmt, welche Tiere sich fortpflanzen und wie lange diese in der Population verbleiben
Eigenschaften	Akkumulation von günstigen Eigenschaften bezogen auf Umwelt	Mensch fördert günstige und verhindert ungünstige
	Wildtiere	Haustiere

Ersatz von NS durch KS ist ein Charakteristikum der Domestikation und eine Ursache der Entstehung der Haustiere aus den Wildtieren

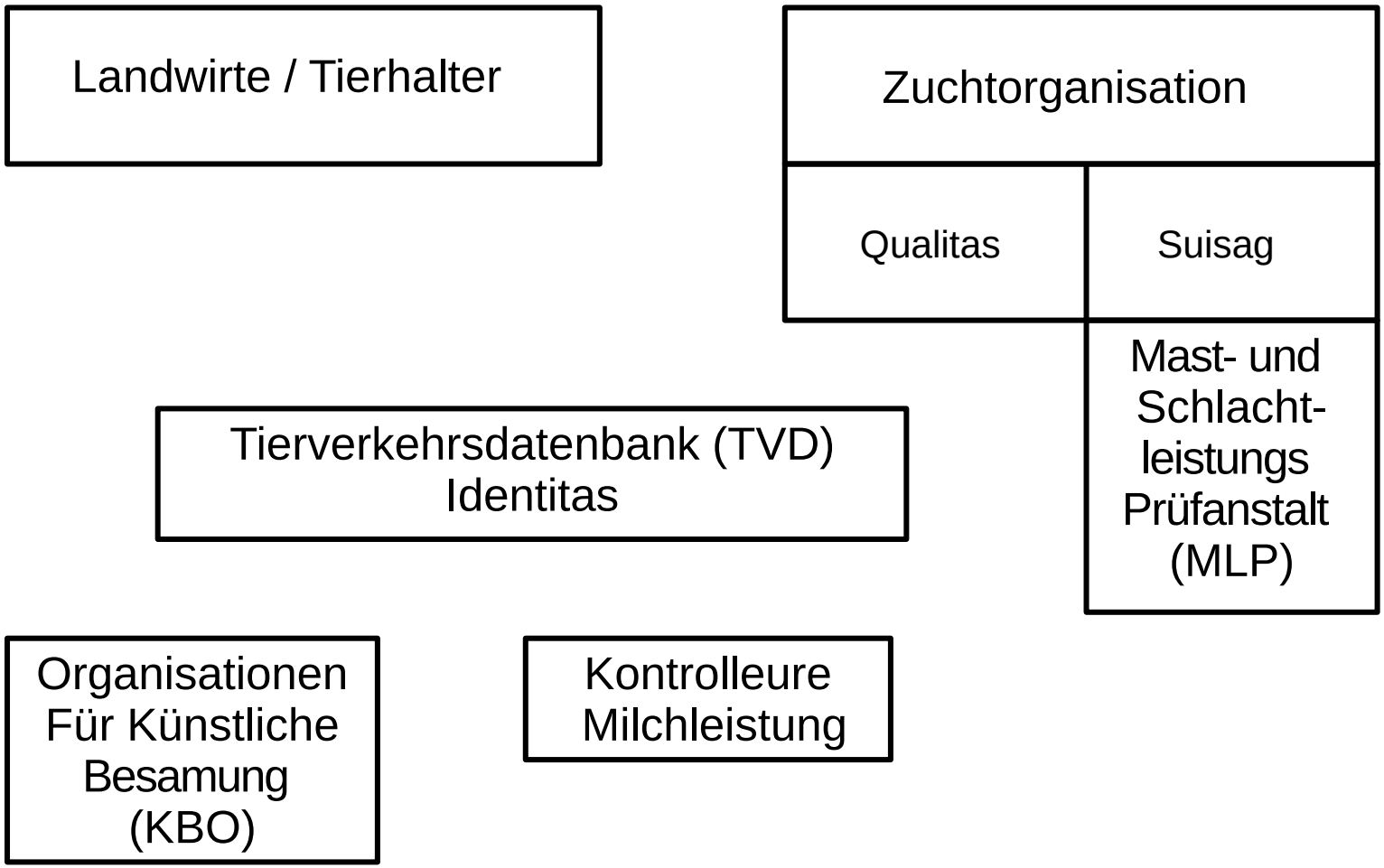
Arten der Paarung

- *Zufällige Paarung* (ZP, Panmixie): Jedes Tier in der Population hat die gleiche Chance sich zu paaren
- *Gerichtete Paarung* (GP, assortative Paarung): Mensch bestimmt, welche Tiere einer Population sich paaren
- KS und gerichtete Paarung sind wichtige Werkzeuge in der Tierzucht

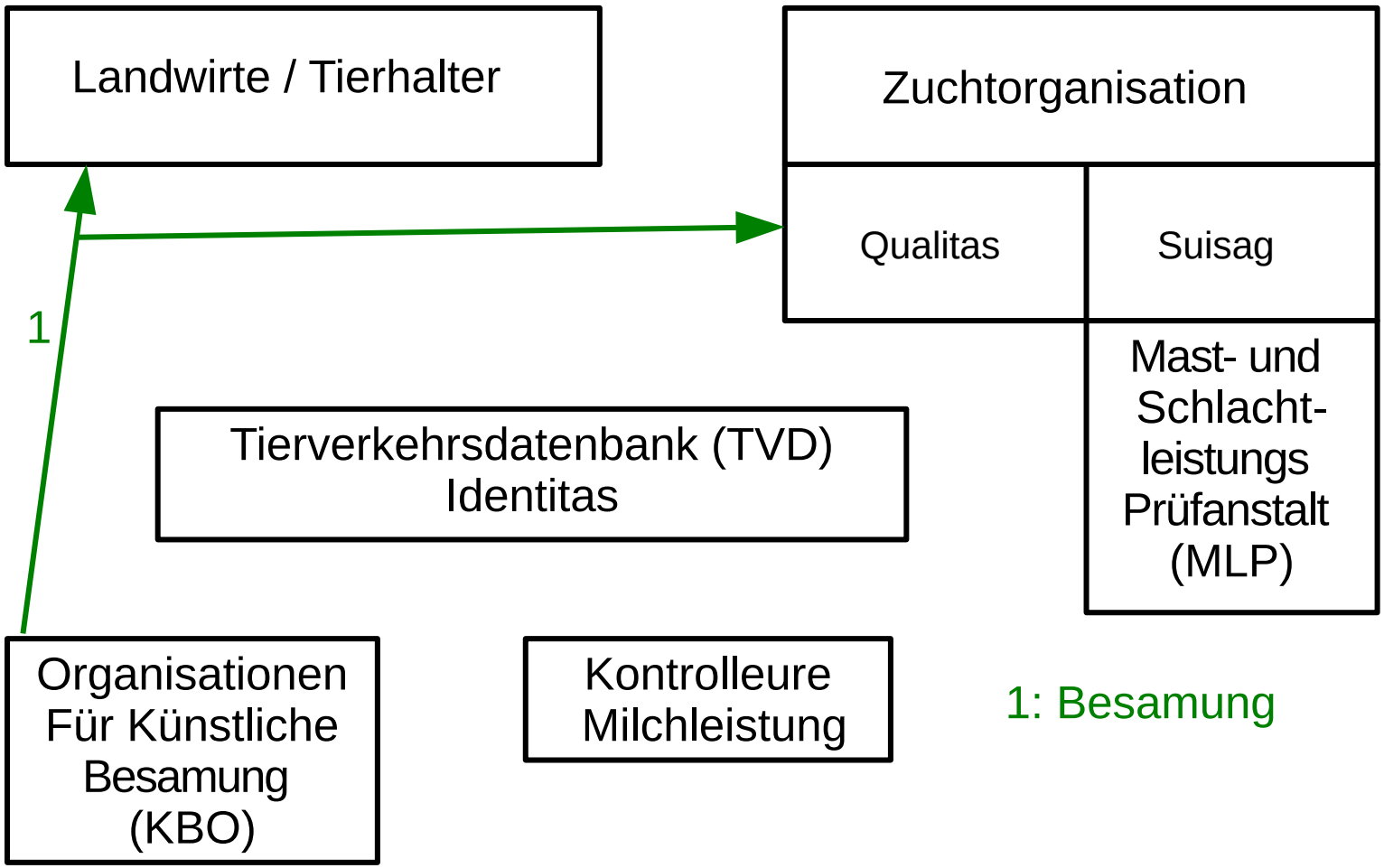
Erfassung der Tiere

- Voraussetzung für die Umsetzung von KS und GP ist eine systematisch Erfassung der Tiere
- Herdebücher als Datenbasis für Abstammung und GP
- Züchtervereinigungen zur Führung der Herdebücher, Definition von Rassestandards und Entwicklung und Umsetzung von Zuchtprogrammen

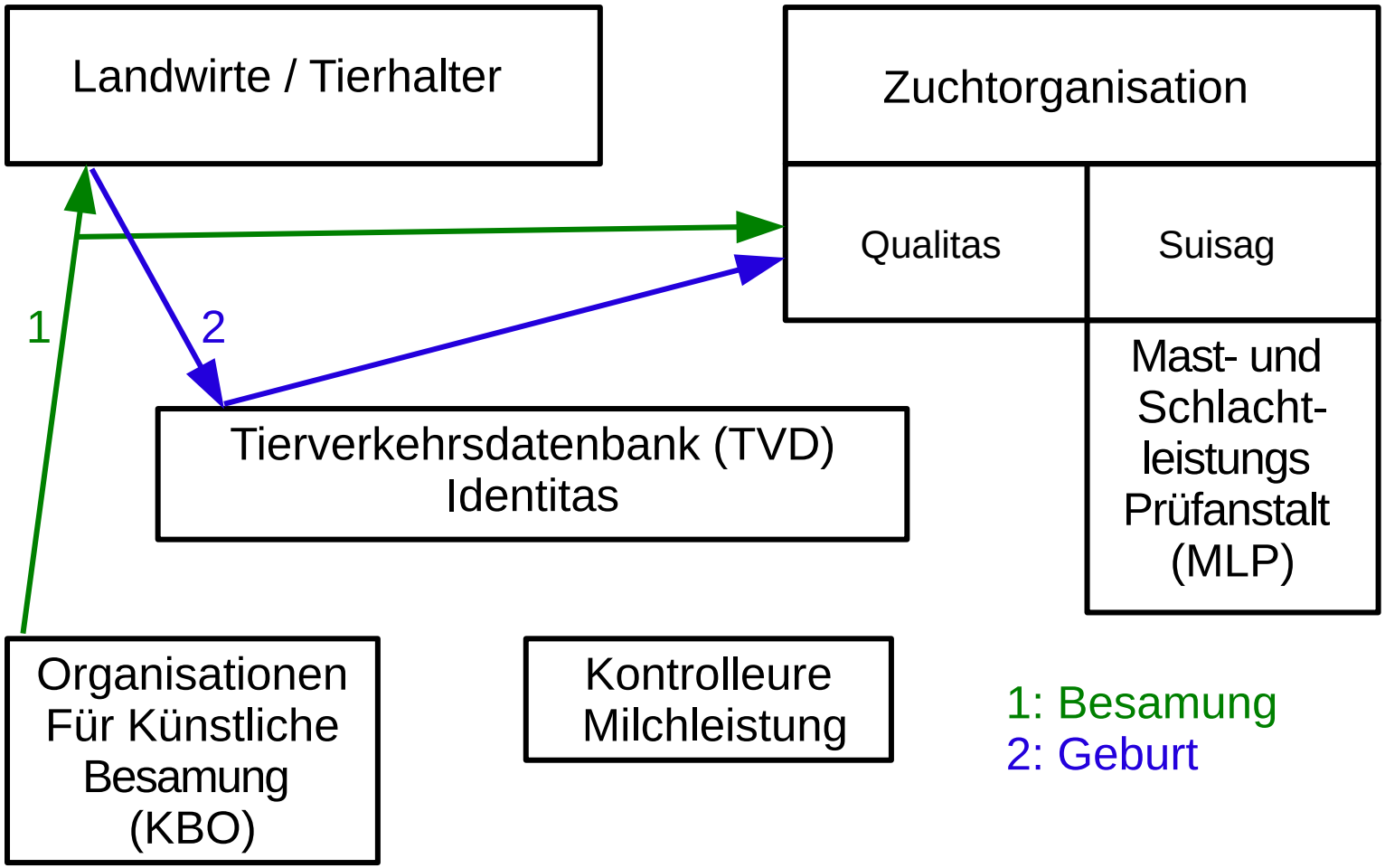
Grundstruktur eines Zuchprogramms



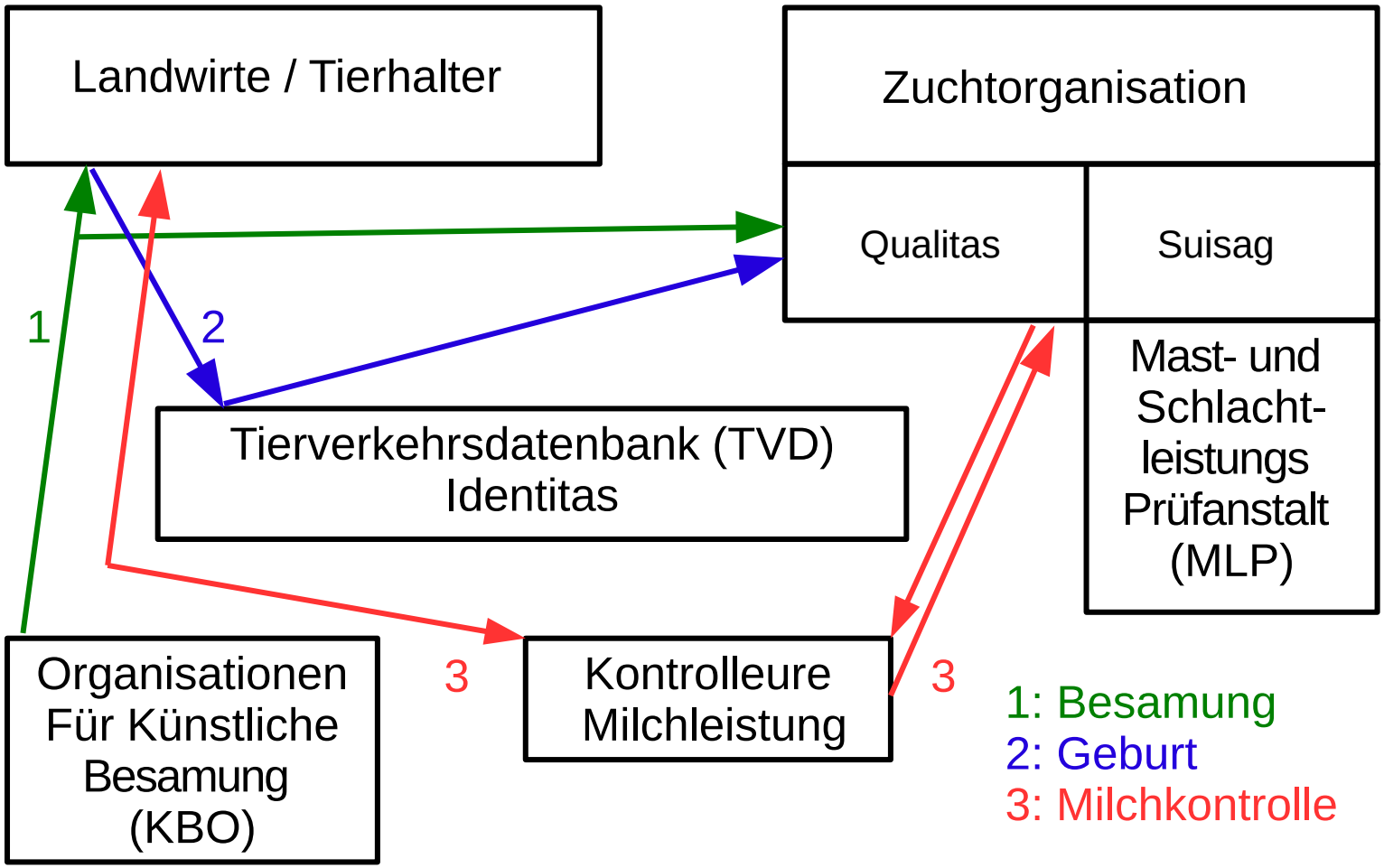
Besamung / Belegung eines weiblichen Tieres



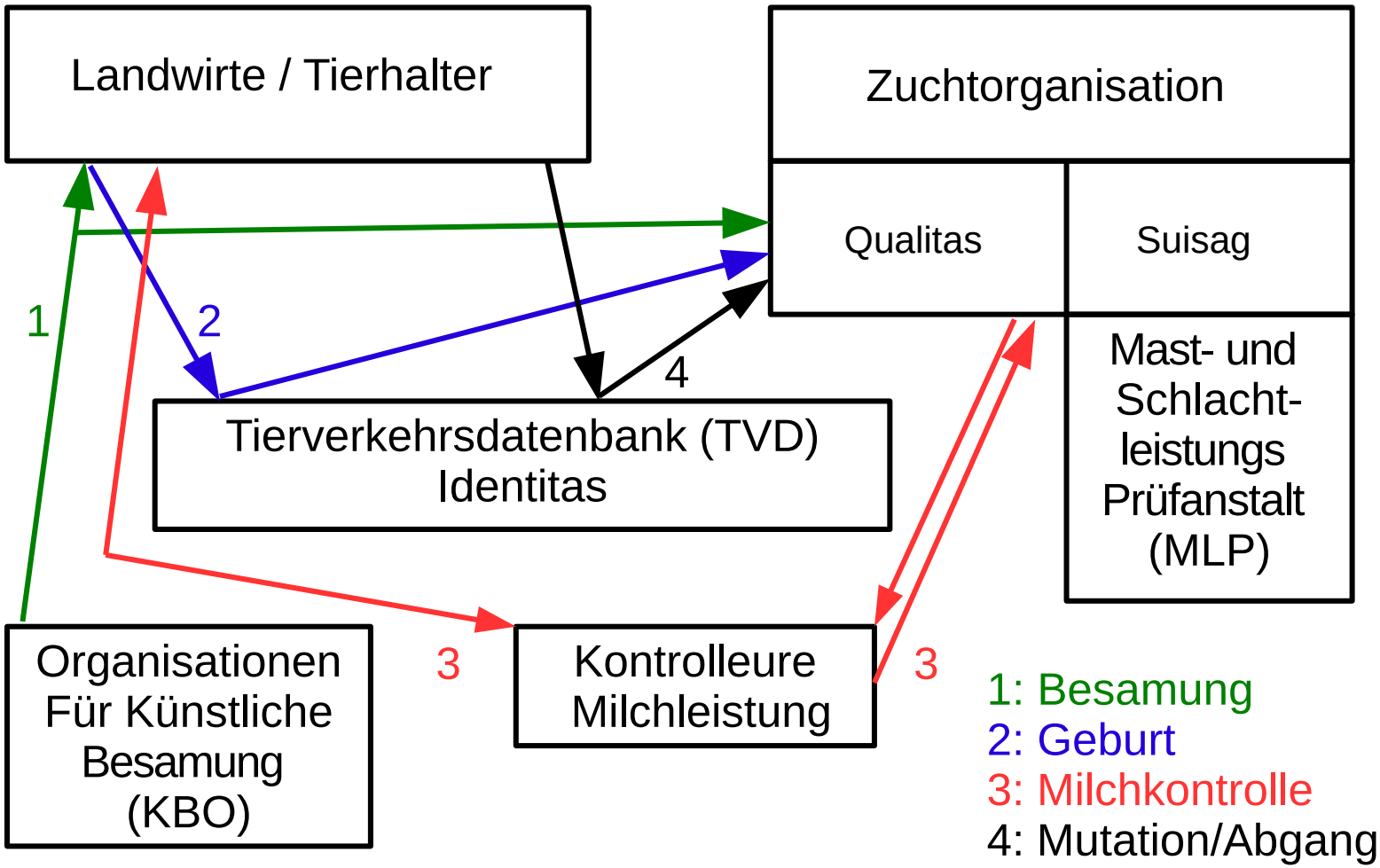
Geburt


Landwirte / Tierhalter
Zuchtorganisation
Qualitas
Suisag
Mast- und Schlacht-leistungs Prüfanstalt (MLP)
1
2
Tierverkehrsdatenbank (TVD) Identitas
Organisationen Für Künstliche Besamung (KBO)
Kontrolleure Milchleistung
1: Besamung
2: Geburt

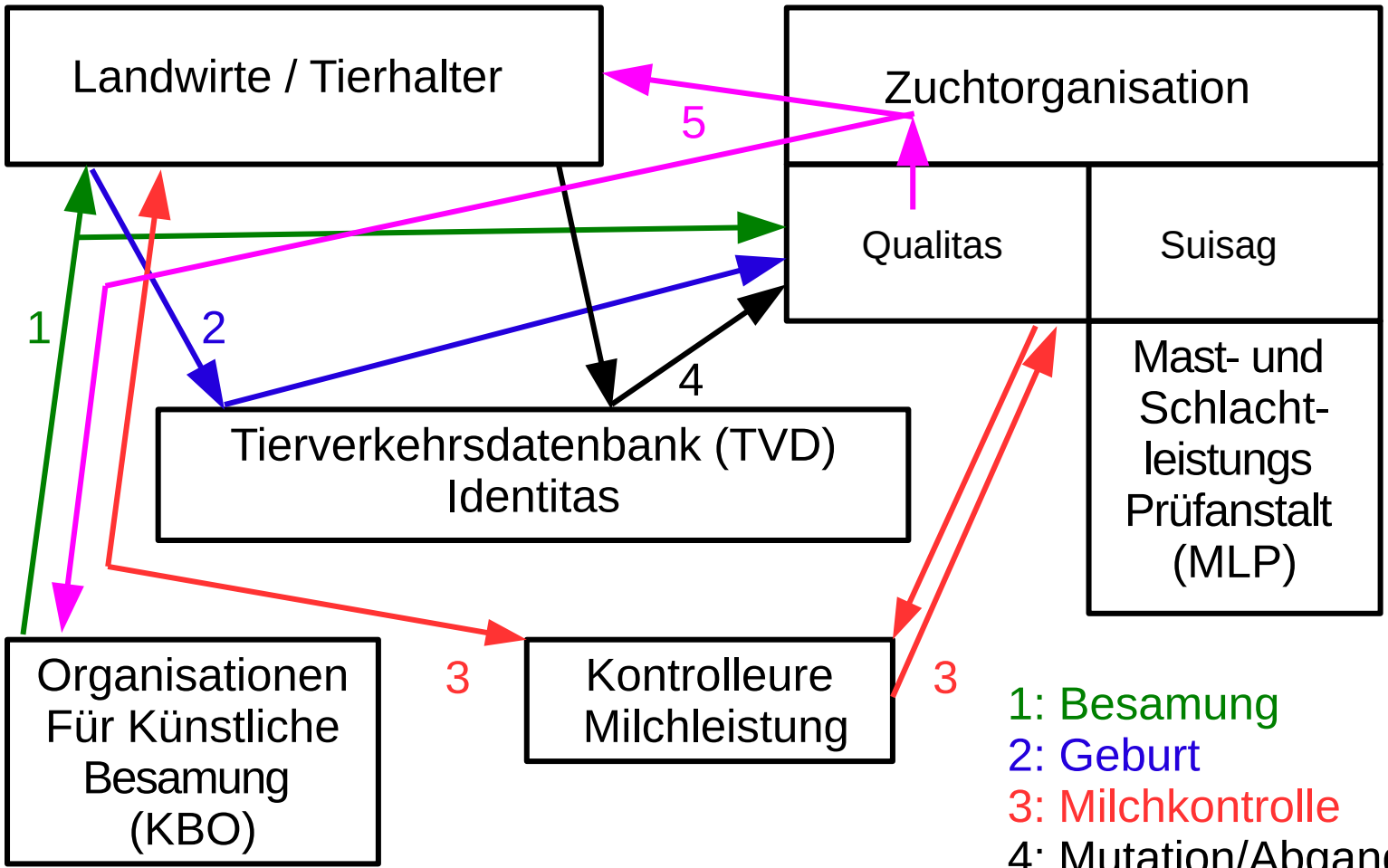
Leistungskontrolle



Mutationen und Abgänge von Tieren

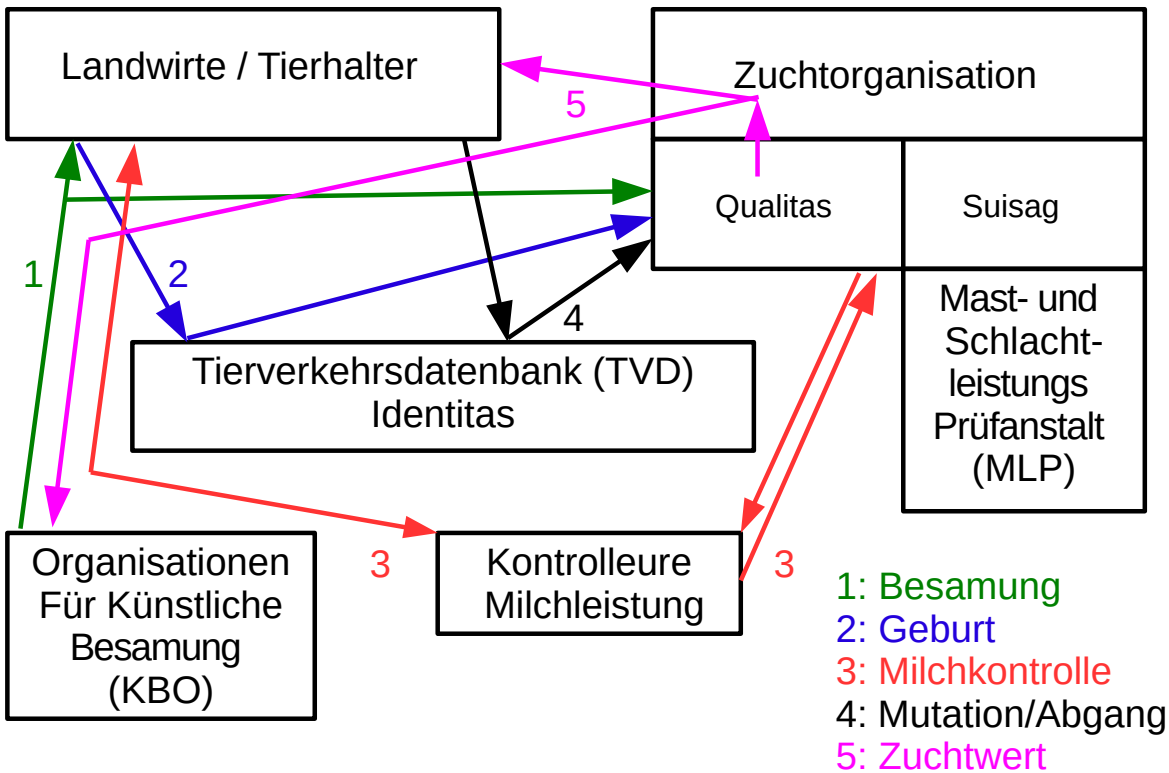


Datenflüsse in einem Zuchtprogramm

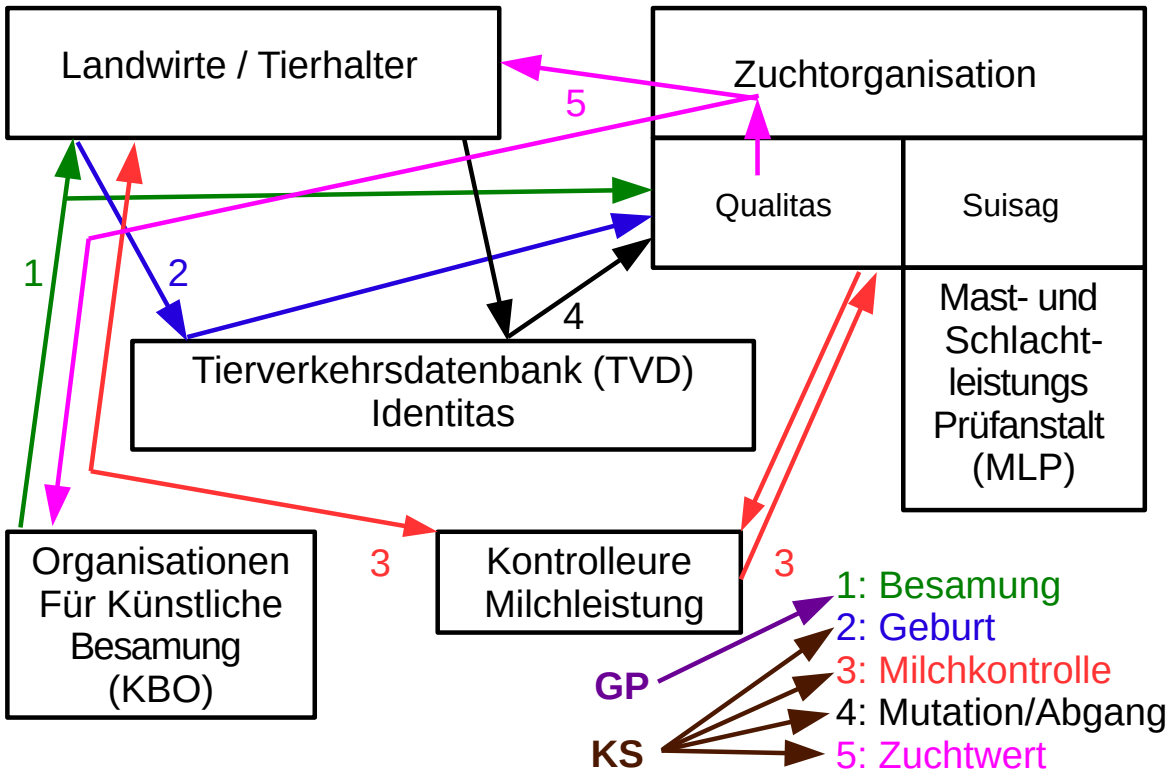


Wo fallen Entscheide betreffend

- Gezielter Paarung (GP)
- Künstlicher Selektion (KS)



Entscheide betreffend GP und KS



Genauere Betrachtung der künstlichen Selektion

Gezielte Paarung (GP)

- GP wichtig an einem Punkt, bei der Besamung/Belegung
- GP ist einfach zu verstehen

Künstliche Selektion (KS)

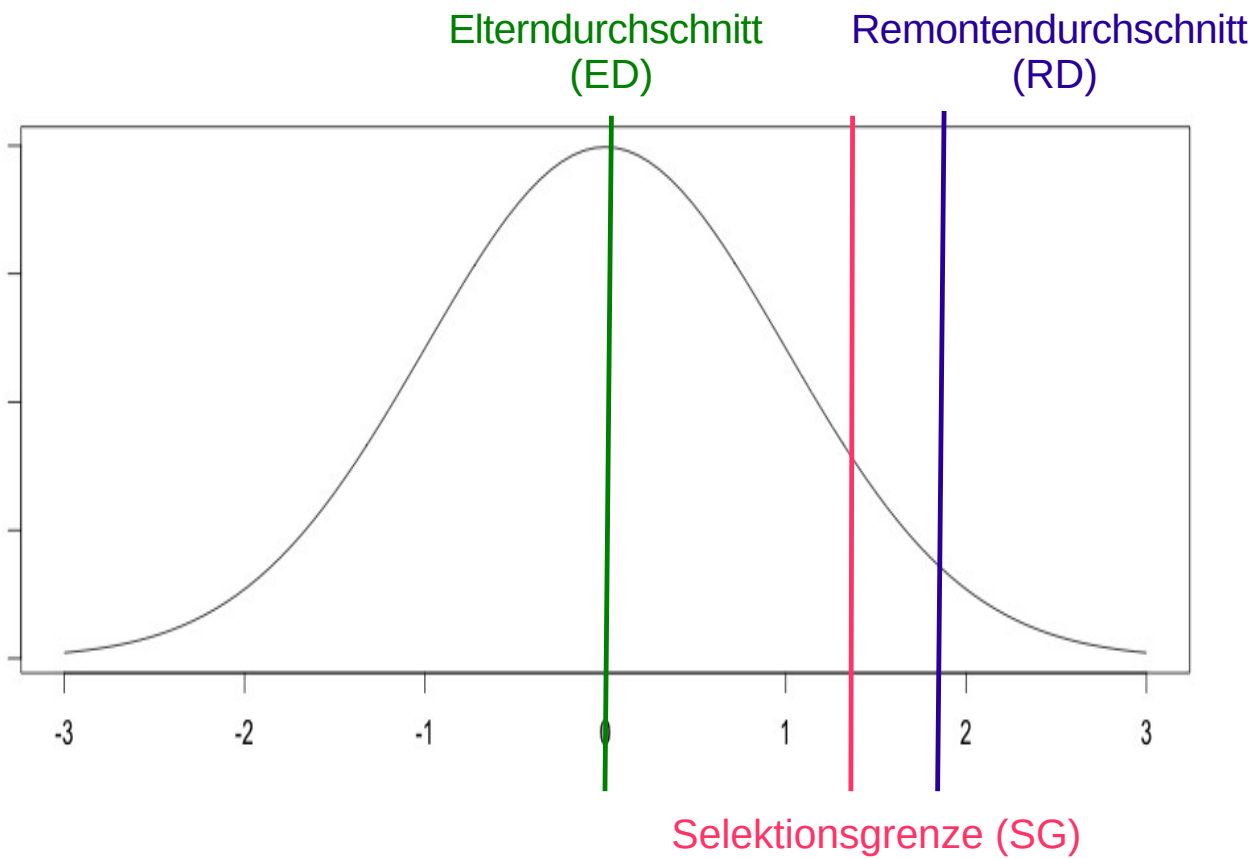
- KS verdient eine genauere Betrachtung
- KS in Tierzucht definiert als gezielte Auswahl von Tieren (werden als *Zuchttiere* bezeichnet) als Eltern der nächsten Generation
- ZüchterIn bestimmt, wie lange Tiere als Zuchttiere im Einsatz sind

Charakteristika der künstlichen Selektion

- *Ziel*: gezielte Veränderung der genetischen Basis der Zuchtpopulation im Sinne des Zuchtziels
- *Effekt*: Änderung der Allelfrequenzen, wobei Frequenz von erwünschten Allelen erhöht und Frequenz von unerwünschten Allelen vermindert
- *Wirkung*: Veränderung des Populationsdurchschnitts von einer Elterngeneration zur Nachkommengeneration wird als Mass der Wirkung verwendet

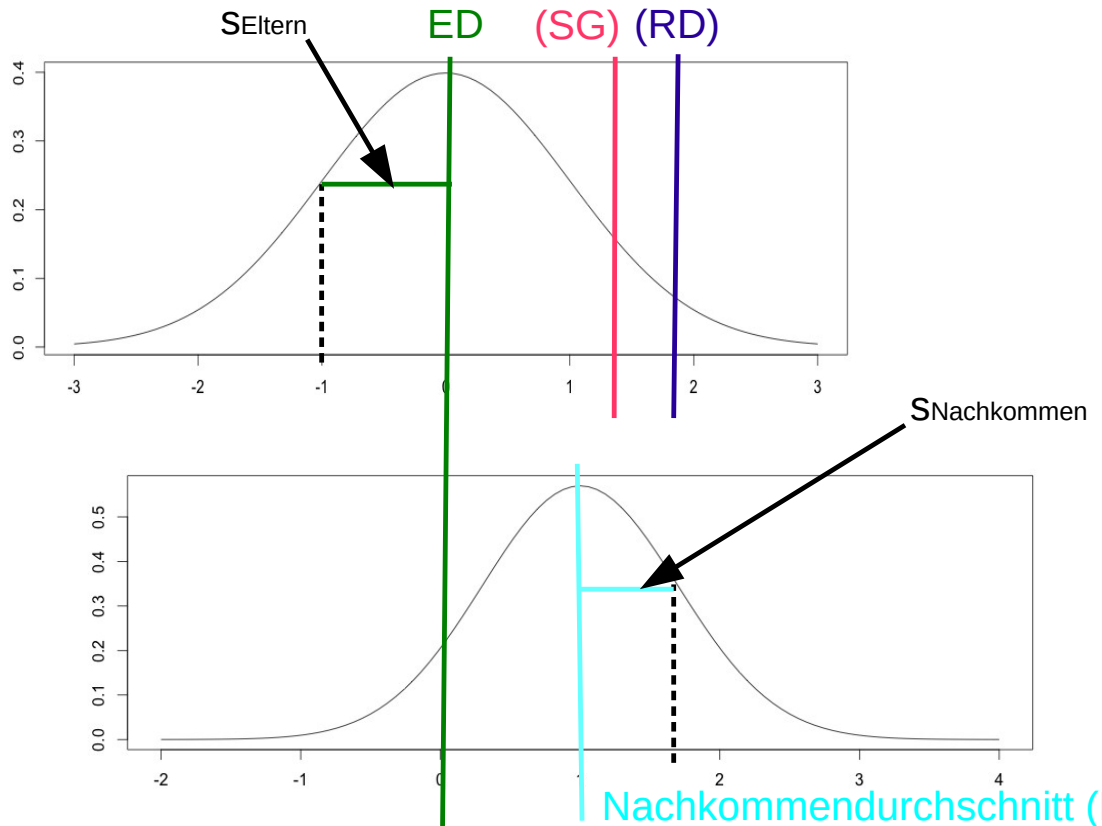
Selektionsformen - Gerichtete Selektion

- Gerichtete Selektion: Erhöhung (oder Verminderung) des Populationsdurchschnitts im Sinne des Zuchtziels



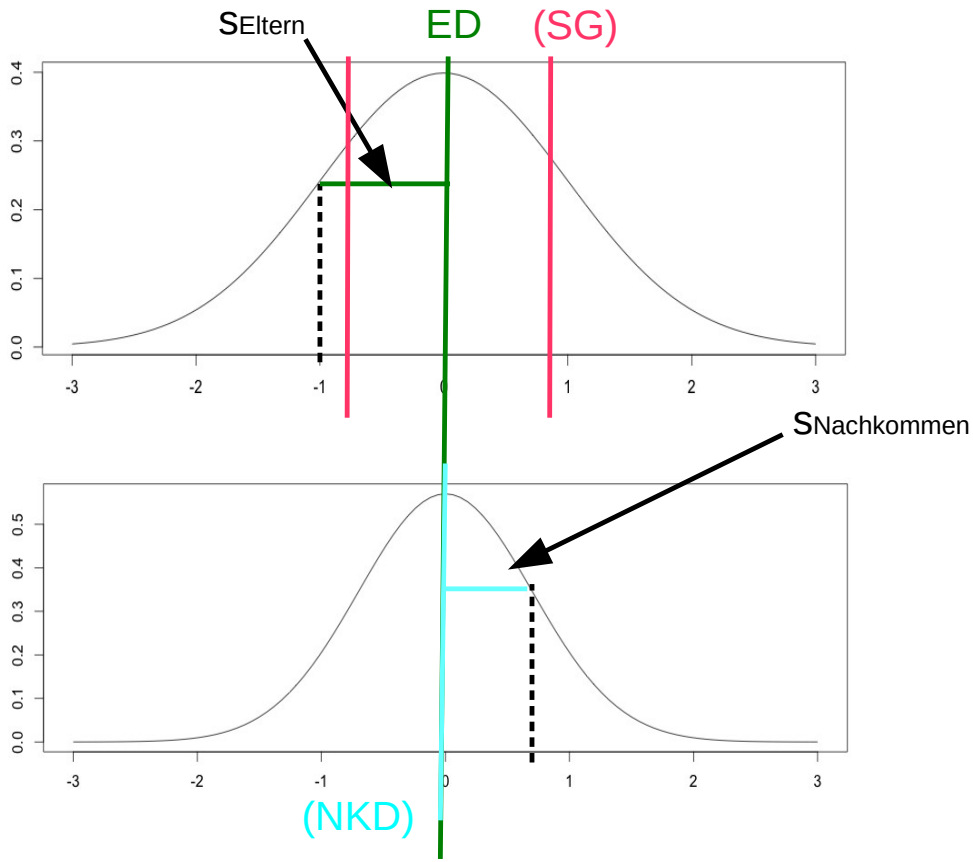
Vergleich von Eltern und Nachkommen bei gerichteter Selektion

- $NKD > ED$, aber $NKD < RD$
- $s_{NK} < s_E$



Selektionsformen - Stabilisierende Selektion

- Stabilisierende Selektion: Populationsdurchschnitts auf bestimmtem Niveau halten
- Beispiel: Optimum-Merkmale wie Intra-Muskulärer Fettgehalt



Erweiterung der Selektion auf mehrere Merkmale

- Bisher gezeigtes Material, alles nur für ein Merkmal - wenig relevant für praktische Anwendung
- Praxis: eine Vielzahl von Merkmalen sind in Zuchtzielen enthalten
- Merkmale in antagonistischer Beziehung, z.Bsp Milchleistung - Fruchtbarkeit
- Wie können Informationen von potentiellen Elterntieren kombiniert und verglichen werden?

Vergleich zweier Tiere mit eigener Leistung

	Kuh Hilda	Kuh Frieda
Milchleistung (kg)	9000	6000
Milchfett (%)	3.5	4.5
Milcheiweiss (%)	3.2	3.5

- Welche Kuh ist besser geeinget als Mutter?
- Wie kann das genetische Potential der beiden Kühe abgeschätzt werden?
- Welcher Massstab soll verwendet werden für die Beurteilung?

Festlegung einer Zielgrösse - Wirtschaftlichkeit

- Definition des Zuchtziels so, dass potentielle Eltern die Wirtschaftlichkeit der Nachkommengeneration verbessern
- Wie kann ich Wirtschaftlichkeit abschätzen?
- $\rightarrow$ Veränderung des Gewinns bei marginaler Veränderung eines Merkmals
- $\rightarrow$ *Wirtschaftliches Gewicht*

Genetisches Potential - Gesamtzuchtwert

- Quantifizierung der Änderung von Eltern zu Nachkommen
- $\rightarrow$ Zuchtwert (siehe Quantitative Genetik)
- Kombination aus wirtschaftlichen Gewichten und Zuchtwerten führt zum *Gesamtzuchtwert*
- **Wichtig**: Gesamtzuchtwert entspricht der mathematischen Formulierung des Zuchtziels

Berechnung Gesamtzuchtwert

- Mathematisch ist der Gesamtzuchtwert **H** als gewichtetes Mittel aus wirtschaftlichen Gewichten **v** und Zuchtwerten **g** definiert
- In Vektorschreibweise heisst das:

$$\mathbf{H} = \mathbf{v}^T * \mathbf{g}$$

- Zurück zu unserem Beispiel

	Hilda (**g**)	Frieda (**g**)	**v**(Fr/Einheit)
Milchleistung (kg)	9000 (+50)	6000 (+150)	+0.05
Milchfett (%)	3.5 (−0.1)	4.5 (+0.05)	−0.01
Milcheiweiss (%)	3.2 (+0.1)	3.5 (+0.1)	+0.005
H (Fr)			

Woher kommen die Zahlen?

- Wirtschaftliche Gewichte **v** können aufgrund von Gewinngleichungen oder aufgrund anderer Methoden berechnet werden(siehe später in dieser Veranstaltung)
- Wahre Zuchtwerte sind unbekannt und werden aus Beobachtungen geschätzt
- Heute: statistische Verfahren zur Schätzung von Zuchtwerten aufgrund von
  - phänotypischen Leistungen
  - Pedigreeinformationen
  - Umweltbedingungen
  - neu: genomische Informationen

Geschichte der Indexselektion

- Hazel und Lush (1943): Gewichtung von phänotypischen Informationen von Individuen zu einem Index **I**, damit die Korrelation r_{IH} zwischen Gesamtzuchtwert **H** und Index **I** maximal ist
- Index **I** sei definiert als

$$\mathbf{I} = \mathbf{b}^T * \mathbf{p}$$

Herleitung der Indexgewichte

- Aus der Definition von **I** und der Bedingung, dass r_{IH} maximal ist, folgt
- $\rightarrow$ $\mathbf{Gv} = \mathbf{Pb}$ (Herleitung für Interessierte)
- $\rightarrow$ $\mathbf{b} = \mathbf{P}^{-1}\mathbf{Gv}$
- **G**: genetische Co-Varianz Matrix
- **P**: phänotypischen Co-Varianz Matrix

Ausblick

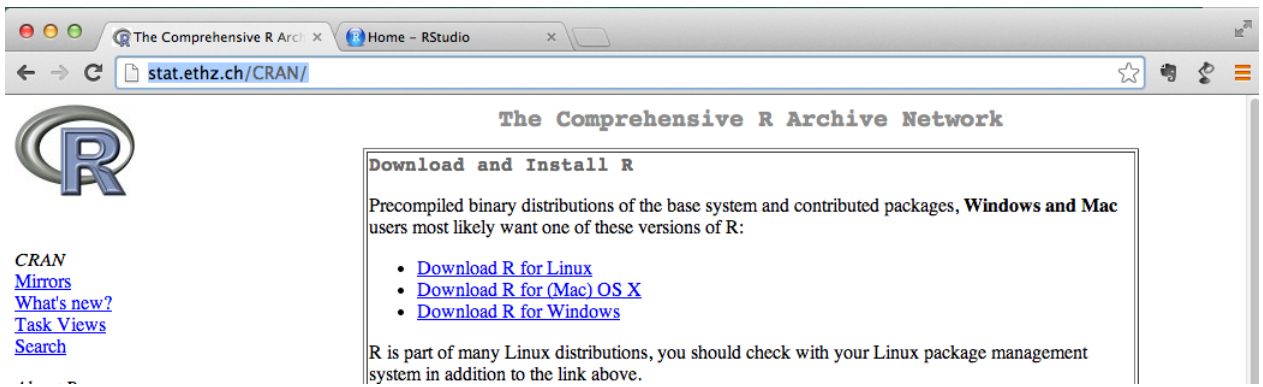
- Anstelle von phänotypischen Eigenleistungen als Informationen kann die Ableitung der Indexgleichung auf andere Informationsquellen erweitert werden.
- Prinzip des Aufstellens der Gleichungen und der Herleitung der Gewichte bleiben gleich.
- Weitere Beispiele für Selektionsindices werden folgen ...

Einführung in R - Wieso R

- Interpretierte Sprache, d.h. ein Programm - der sogenannte Interpreter - wartet auf Eingaben
- Einfache Installation: Download und Installation, analog zu anderen Programmen
- Verfügbarkeit: Windows, Mac, Linux
- Gratis und offen, d.h. Programm-code ist verfügbar
- Gute Unterstützung durch externe Tools, z. Bsp RStudio
- Grosse Benutzergemeinde, da sehr populär in Statistik
- Flexibles Arbeiten und schnelles Prototyping, d.h. Ideen sind sehr schnell in Programmcode verwandelt
- Gute Grafikfähigkeiten, d.h. Daten können schnell und einfach visualisiert werden

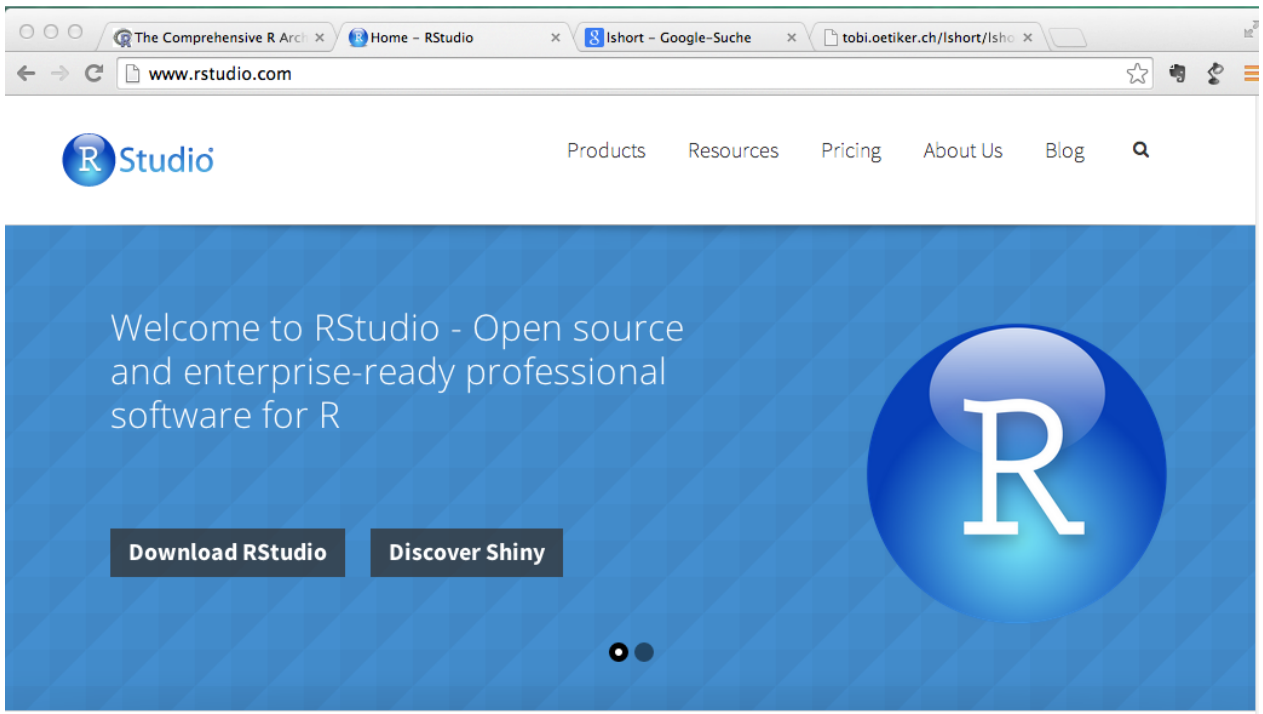
Installation

- Download von CRAN (Comprehensive R Archive Network): http://stat.ethz.ch/CRAN/
- Installation des Binaries (.exe unter Windows und .pkg unter Mac) mit Doppelclick auf Datei
- Webseite von R: https://www.r-project.org/



Installation von RStudio

- Download von https://www.rstudio.com
- Installation des Binaries (verlangt, dass R schon installiert ist)



Erste Sitzung mit R - Interaktiver Modus

- R kann wie ein Taschenrechner verwendet werden $\hat{=}$ *interaktiver Modus*

```
> 5 + 7

[1] 12

> 9*3

[1] 27
```

- Es gilt Klammer vor Punkt vor Strich

```
> 45 - 15 / 2

[1] 37.5

> (45 - 15) / 2

[1] 15
```

Erste Sitzung mit R - Mehr Operatoren

- Quadratwurzel

```
> sqrt(2)
```

```
[1] 1.414214
```

- Potenz

```
> 4.92 ^ 3
```

```
[1] 119.0955
```

- Logarithmus

```
> log(15)
```

```
[1] 2.70805
```

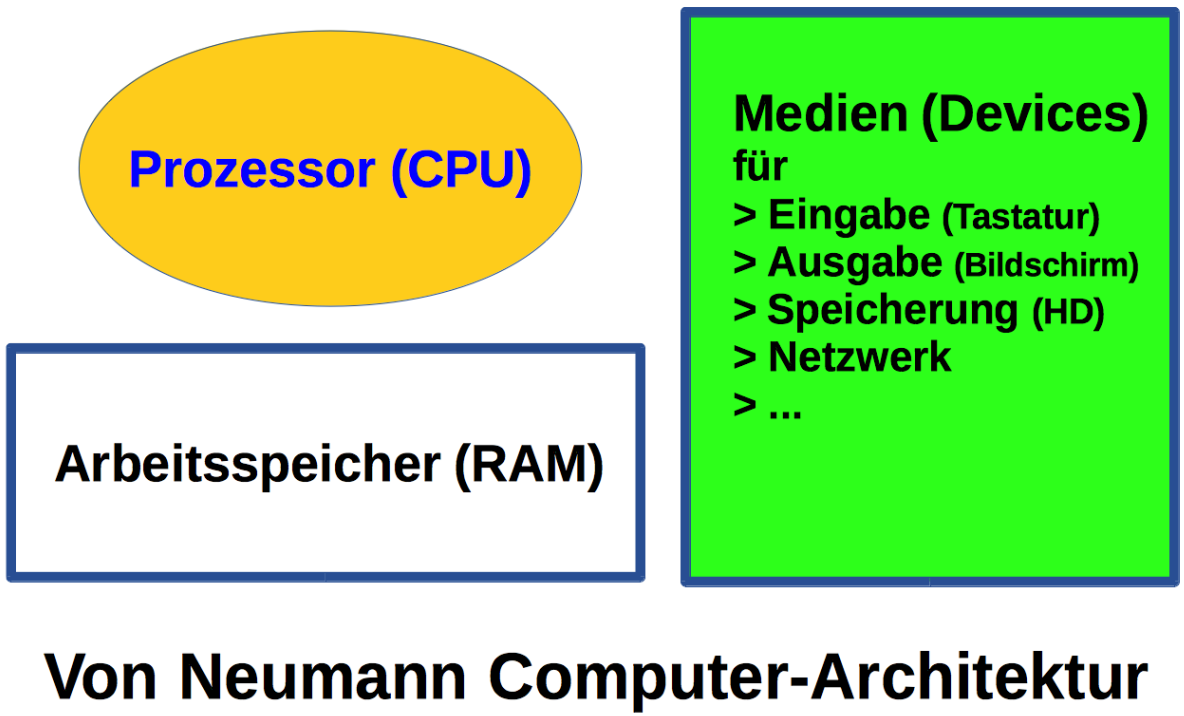
Variablen = Objekte

- Interaktiver Modus mühsam, falls längere Berechnungen mit Zwischenergebnissen
- Beispiel: Mittelwert m und Standardabweichung s von fünf Zahlen
- Mittelwert: Interaktiv

```
> (15 + 9 + 8 + 34 + 76) / 5

[1] 28.4

> sqrt(((15-28.4)^2 + (9-28.4)^2 + (8-28.4)^2
+         + (34-28.4)^2 + (76-28.4)^2)/4)

[1] 28.58846
```

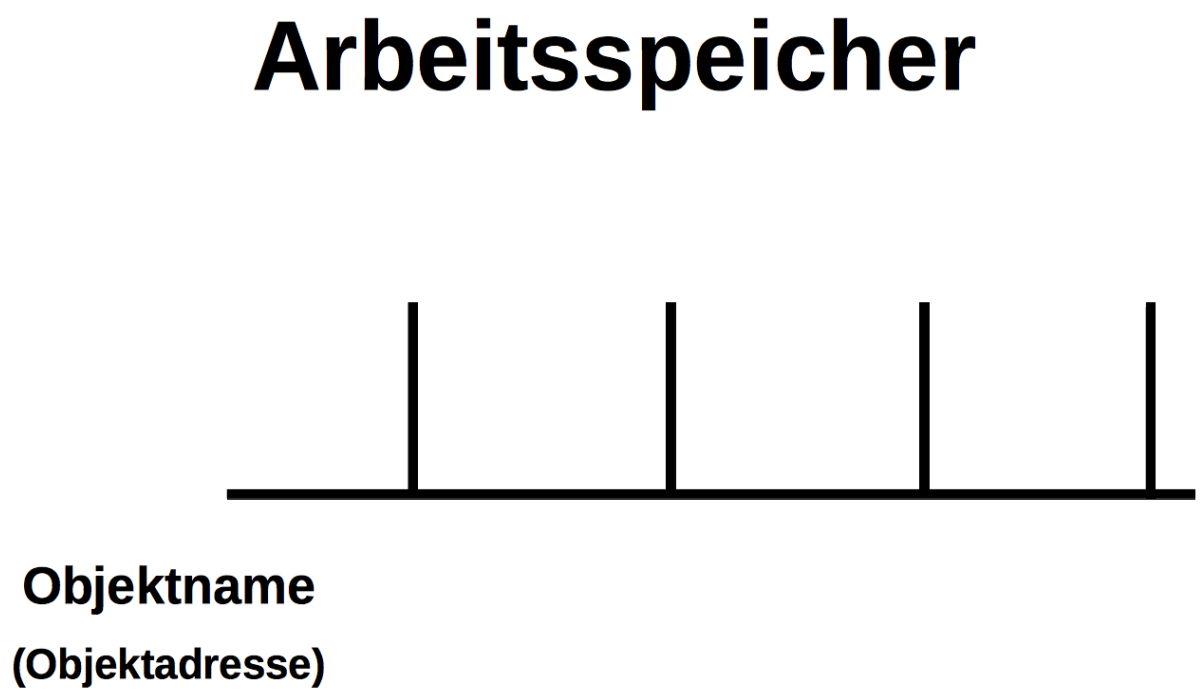
Variablen = Objekte II

- Verbesserung: Ablage von Zwischenresultaten im Arbeitsspeicher
- Wie sieht das aus im Arbeitsspeicher eines Computers?



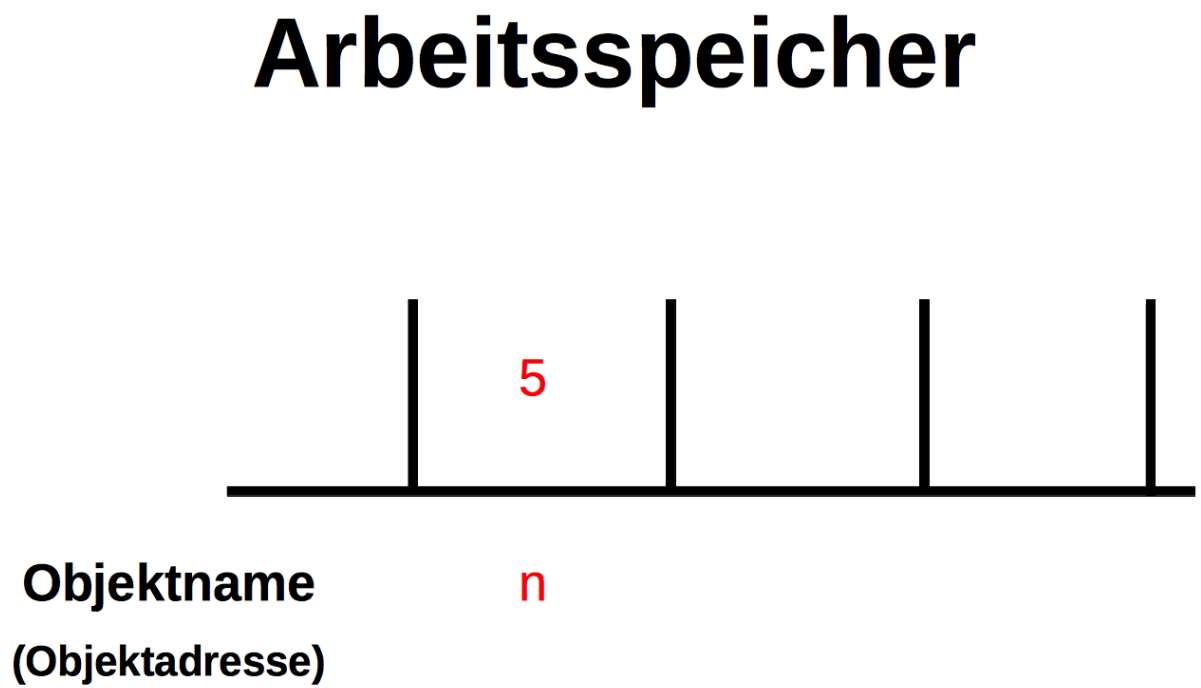
Von Neumann Computer-Architektur

Arbeitsspeicher



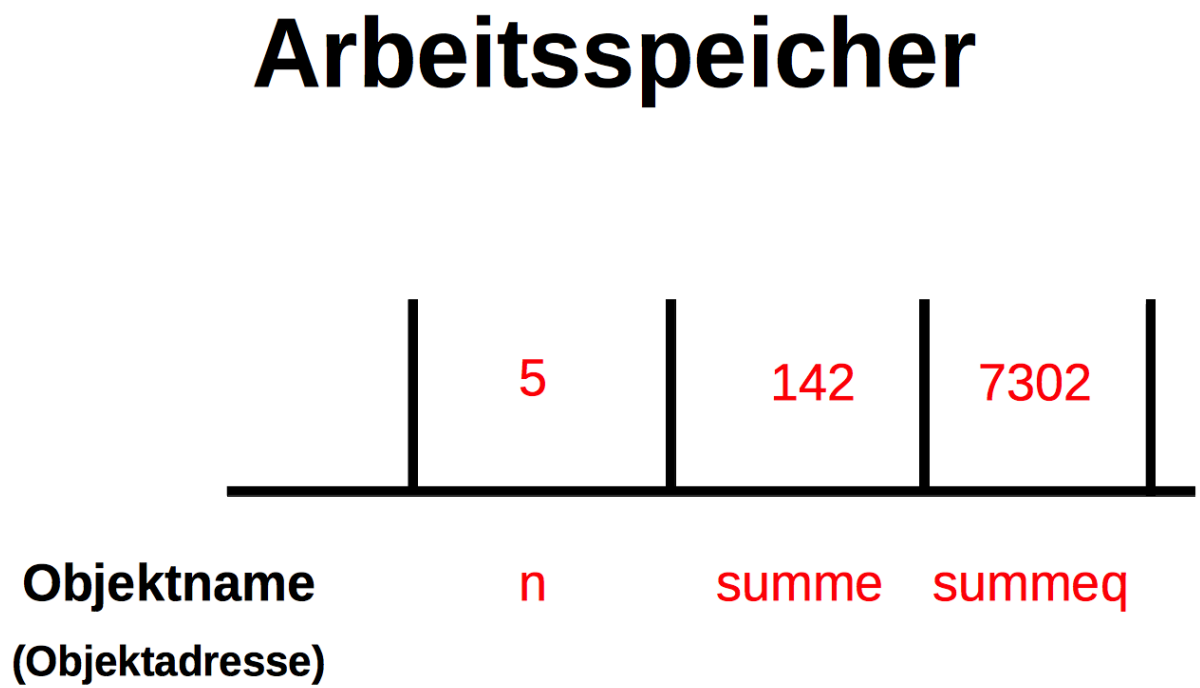
Zuweisung von Werten zu Objekten

```
> n <- 5
```



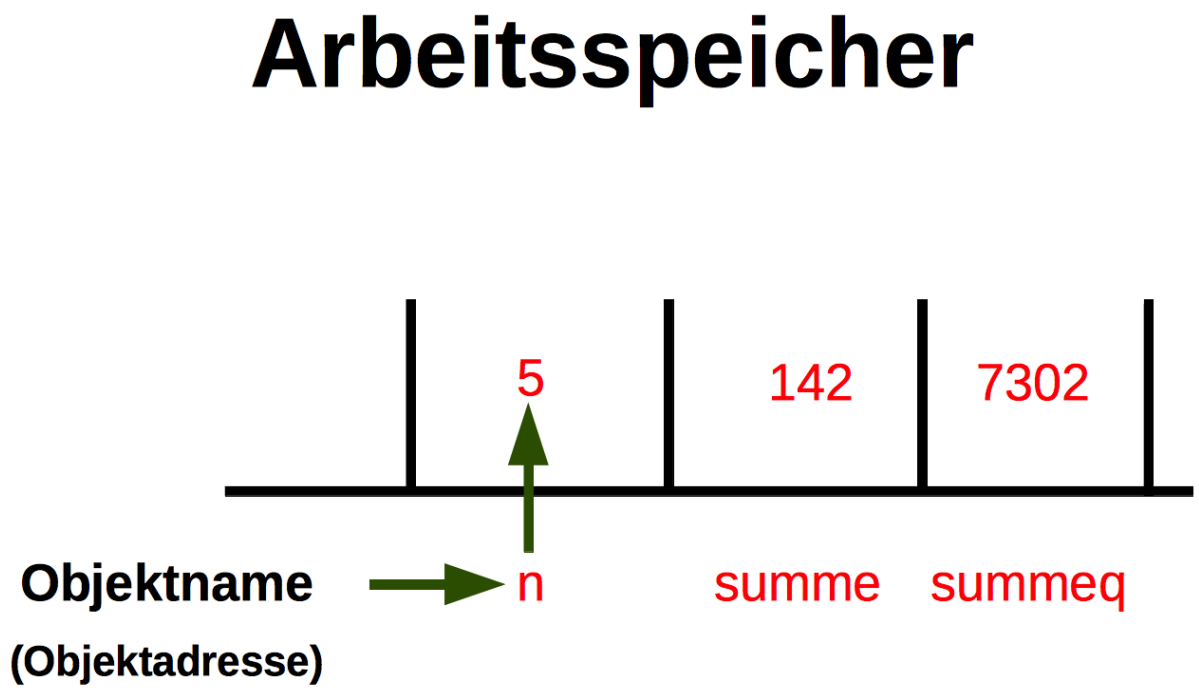
Zuweisung von Werten zu Objekten II

```
> summe <- 15 + 9 + 8 + 34 + 76
> summeq <- 15^2 + 9^2 + 8^2 + 34^2 + 76^2
```



Zugriff auf Werte

```
> n
[1] 5
```



Namen von Objekten

- Kann Buchstaben, Zahlen und Zeichen wie "-", "_" oder "." enthalten
- Nicht mit einer Zahl beginnen
- Reservierte Namen wie c, i, t, oder andere Funktionsnamen nicht als Objektnamen verwenden
- Allgemeine Hinweise zu Style, Namen und Notationen: http://r-pkgs.had.co.nz/style.html

Rechnen mit Objekten

- Mittelwert m

```
> m <- summe / n
```

- Standardabweichung s

```
> s <- sqrt((summeq - summe^2/n)/(n-1))
```

- Der R-interpreter gibt bei Zuweisungen kein Resultat zurueck (ausser man macht eine Klammer)
- Output wird angezeigt, wenn Objektname eingegeben wird oder mit der `print()`-Funktion

```
> m
```

```
[1] 28.4
```

```
> print(s)
```

```
[1] 28.58846
```

Erzeugung von Objekten mit Funktionen

- Bisher: Objekte durch Zuweisungen erzeugt
- Funktionen `sqrt()` und `print()` schon angetroffen
- Beispiel - Vektor

```
> (vecNum <- vector(mode = "numeric", length = 2))
[1] 0 0
```

- Hilfe zu einer Funktion mit

```
> help("vector")
> ?vector
```

Datentypen in R

Es werden sechs Datentypen unterschieden

1. `numeric`: reelle Zahlen, default für Zahlen in R
2. `integer`: ganze Zahlen (0, 1, 2, ...) mit "L" hinter der Zahl
3. `complex`: Quadratwurzeln aus negativen Zahlen
4. `character`: Buchstaben, Zeichen, ...
5. `factor`: spezieller Datentype für lineare Modelles

Wichtige Punkte zu Datentypen

Datentypen - Prüfung und Umwandlung

- R kennt keine strenge Prüfung von Datentypen (strong typing)
- R konvertiert Datentypen automatisch (coersion)
- Coersion ist bequem, kann aber auch Fehlerquelle sein

Hilfreiche Funktionen

- `class()` gibt den Type eines Objekts zurück
- `is.<data.type>()` prüft, ob ein Objekt von gewissen Datentyp ist, z. Bsp `is.integer(5)`
- `as.<data.type>()` kann für eine explizite Umwandlung verwendet werden z. Bsp `as.character(12)`

Vektoren

- Ein Vektor ist eine Sequenz von Datenelementen vom gleichen Basistyp
- Datenelemente in einem Vektor werden als Komponenten bezeichnet
- In R werden Vektoren mit der Funktion `vector()` erzeugt und mit Funktion `c()` erweitert.
- Beispiel:

```
> vecNum <- vector(mode = "numeric", length = 2)
> vecNum[1] <- 5
> vecNum[2] <- -1
> print(vecNum)
[1]  5 -1
```

Vektoren II

- ... oder in einem Schritt mit `c()`

```
> vecNum <- c(5,-1)
> print(vecNum)
```

```
[1]  5 -1
```

- Erweiterung bestehender Vektoren mit

```
> vecChar <- c("aa","bb","da")
> vecChar <- c(vecNum, vecChar)
> print(vecChar)
```

```
[1] "5"  "-1" "aa" "bb" "da"
```

Eigenschaften von Vektoren

- Anzahl Komponenten in einem Vektor

```
> length(vecChar)

[1] 5
```

- Test ob ein Vektor leer ist, also keine Elemente enthält

```
> length(vecChar) == 0

[1] FALSE

> vecLogical <- vector(mode = "logical", length = 0)
> length(vecLogical) == 0

[1] TRUE
```

Logische Operationen mit Vektoren

- Logische oder Boolsche Operationen (grösser, kleiner, gleich) auf Vektoren werden elementweise ausgeführt
- Resultat entspricht einem Vektor der gleichen Länge wie der ursprüngliche Vektor mit TRUE und FALSE Werten
- Beispiel:

```
> a <- c(33,5,7,13,-1)
> a > 5
```

```
[1]  TRUE FALSE  TRUE  TRUE FALSE
```

Vektorarithmetik - Plus, Minus

- Arithmetische Operationen mit Vektoren (plus, minus, mal, durch) werden elementweise ausgeführt
- Beispiel

```
> x <- c(3,5,13,-2)
> y <- c(2,6,-3,19)
> x+y
[1]  5 11 10 17
> x-y
[1]   1  -1  16 -21
```

Vektorarithmetik - Mal, Durch

- Elementweise Ausführung analog zu Plus, Minus

```
> x*y
[1]   6  30 -39 -38
> x/y
[1]  1.5000000  0.8333333 -4.3333333 -0.1052632
```

- Skalarprodukt zwischen zwei Vektoren mit

```
> crossprod(x,y)
     [,1]
[1,]  -41
```

Zugriff auf Komponenten im Vektor

- Numerischer Index

```
> x[2]

[1] 5
```

- Negativem Indices schliessen Komponenten aus

```
> x[-3]

[1]  3  5 -2
```

- Bereiche von Indices

```
> x[3:4]

[1] 13 -2
```

Matrix

- Matrix = Stapel von mehreren Vektoren gleicher Länge, d.h. Matrizen sind zweidimensionale Objekte, welche in Kolonnen und Spalten organisiert sind
- Komponenten von Matrizen sind vom gleichen Typ
- Beispiel:

```
> matA <- matrix(c(5,3,4,-6,3,76), nrow = 2, ncol = 3,
+                byrow = TRUE)
> print(matA)
     [,1] [,2] [,3]
[1,]    5    3    4
[2,]   -6    3   76
```

Dimension einer Matrix

- Eine grundlegende Eigenschaft einer Matrix ist ihre Dimension
- Dimension = Anzahl Zeilen und Anzahl Kolonnen
- Darstellung als Vektor der Länge zwei
- Beispiel

```
> dim(matA)
[1] 2 3
```

Zugriff auf Matrix Elemente

- Per numerischem Index

```
> matA[2,1]
```

```
[1] -6
```

- Zugriff auf ganze Zeile

```
> matA[1,]
```

```
[1] 5 3 4
```

- Zugriff auf ganze Spalte

```
> matA[,2]
```

```
[1] 3 3
```

Zeilenweise Erweiterung einer Matrix

```
> matB <- matrix(c(3,-1,90,1,1,4), nrow = 2, ncol = 3,
+                byrow = TRUE)
> rbind(matA, matB)
     [,1] [,2] [,3]
[1,]    5    3    4
[2,]   -6    3   76
[3,]    3   -1   90
[4,]    1    1    4
```

Kolonnenweise Erweiterung einer Matrix

```
> cbind(matA, matB)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    5    3    4    3   -1   90
[2,]   -6    3   76    1    1    4
```

Matrix Operationen

- Transponieren: Zeilen werden Kolonnen und Kolonnen werden Zeilen

```
> t(matA)
```

```
     [,1] [,2]
[1,]    5   -6
[2,]    3    3
[3,]    4   76
```

- Transponierte der Transponierten gibt ursprüngliche Matrix

```
> t(t(matA))
```

```
     [,1] [,2] [,3]
[1,]    5    3    4
[2,]   -6    3   76
```

Addition und Subtraktion

- Addition und Subtraktion werden element-weise ausgeführt

```
> matA + matB
```

```
     [,1] [,2] [,3]
[1,]    8    2   94
[2,]   -5    4   80
```

```
> matA - matB
```

```
     [,1] [,2] [,3]
[1,]    2    4  -86
[2,]   -7    2   72
```

Matrix Multiplikation

- Regel: Zeile mal Kolonne
- Dimensionen müssen stimmen

```
> matA %*% t(matB)
```

```
     [,1] [,2]
[1,]  372   24
[2,] 6819  301
```

Liste

- Eine Liste fasst eine Sammlung von Objekten zusammen.
- Einzelne Objekte heissen *Komponenten*
- Komponenten innerhalb der Liste haben eine fixe Reihenfolge
- Komponenten müssen nicht vom selben Typ sein
- Beispiel

```
> lstA <- list(nZahlen = c(5,-2,7),
+              sNamen = c("Fred","Mary"),
+              lBool = c(FALSE,TRUE))
```

Zugriff auf Listenelemente

- Mit Index und einfacher Klammerung, Resultat ist wieder eine Liste

```
> lstA[2]

$sNamen
[1] "Fred" "Mary"
```

- Mit Komponentennamen

```
> lstA["nZahlen"]

$nZahlen
[1]  5 -2  7
```

Zugriff auf Listenelemente II

- Mit Index und doppelter Klammer, Resultat ist Vektor

```
> lstA[[1]]
```

```
[1]  5 -2  7
```

- Mit $ und Komponentennamen

```
> lstA$lBool
```

```
[1] FALSE  TRUE
```

Spezielle Listen - Dataframes

- Ein *Dataframe* ist eine Liste von Vektoren, welche alle die gleiche Länge haben
- Erzeugung mit Funktion `data.frame()`

```
> dfA <- data.frame(nZahl = c(-2,15),
+                   sZeichen = c("Alice","Bob"),
+                   bWahr = c(FALSE,FALSE))
```

- Zugriff, wie bei Matrix mit Indices oder mit Namen

```
> dfA[2,1]
[1] 15
> dfA$nZahl
[1] -2 15
```

Daten in R einlesen

- Eine der wichtigsten Aufgaben ist, dass externe Daten in R eingelesen werden können
- Die wichtigste Funktion für das Einlesen von Daten: `read.table()` oder darauf aufbauende Funktionen, wie z. Bsp `read.csv2()` zum Einlesen von Daten im CSV-Format
- Häufige Prozedur für den Transfer von Daten aus Excel:
  1. Speichern als Text(csv)
  2. Einlesen in R mit `read.csv2()`

Daten in R einlesen II

- Messungen von Brustumfang und Gewicht als Excel-Tabelle
- Einlesen dieser Daten in R

```
> dfBrGew <- read.csv2(file = "br_gew.csv")
> dim(dfBrGew)
```

```
[1] 10  2
```

```
> head(dfBrGew, 3)
```

```
  Brustumfang Gewicht
1         176     471
2         177     463
3         178     481
```

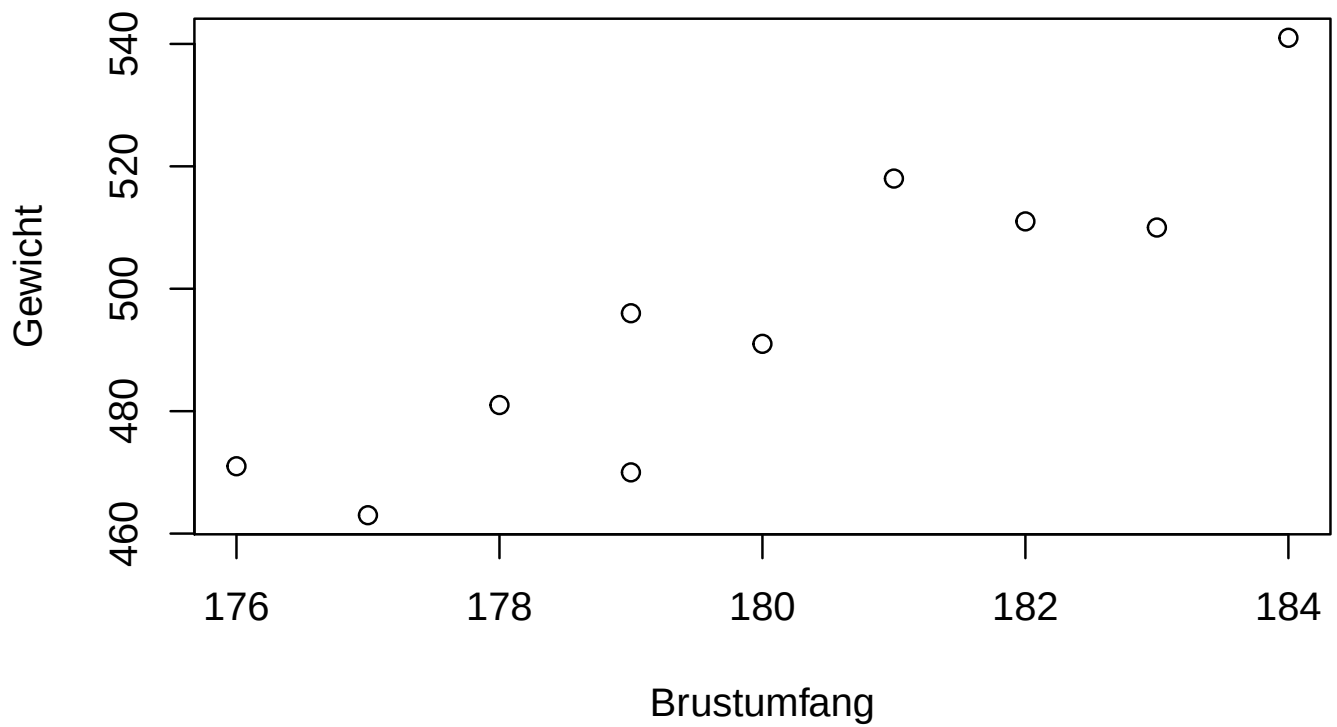
Schreiben von Daten in Dateien

- Dataframes oder Matrizen werden mit der Grundfunktion `write.table()` in Dateien geschrieben
- Analog zu `read.csv2()` gibt es `write.csv2()`
- Für nicht csv-Daten kann `cat()` verwendet werden

Diagramme und Plots

- Plots in R sind einfach: Funktion `plot()`

```
> plot(dfBrGew)
```



Einfaches lineares Modell

- Regressionsmodell von Gewicht auf Brustumfang

```
> lmBrGew <- lm(Gewicht ~ Brustumfang, data = dfBrGew)
> plot(dfBrGew)
> abline(coef = coefficients(lmBrGew), col = "red")
```