Seminar

Anwendungen und Algorithmen basierend auf Prinzipien eines künstlichen Immunsystems

Simulation des Immunsystems

Uwe Schreiner

Betreuer: Thomas Stibor

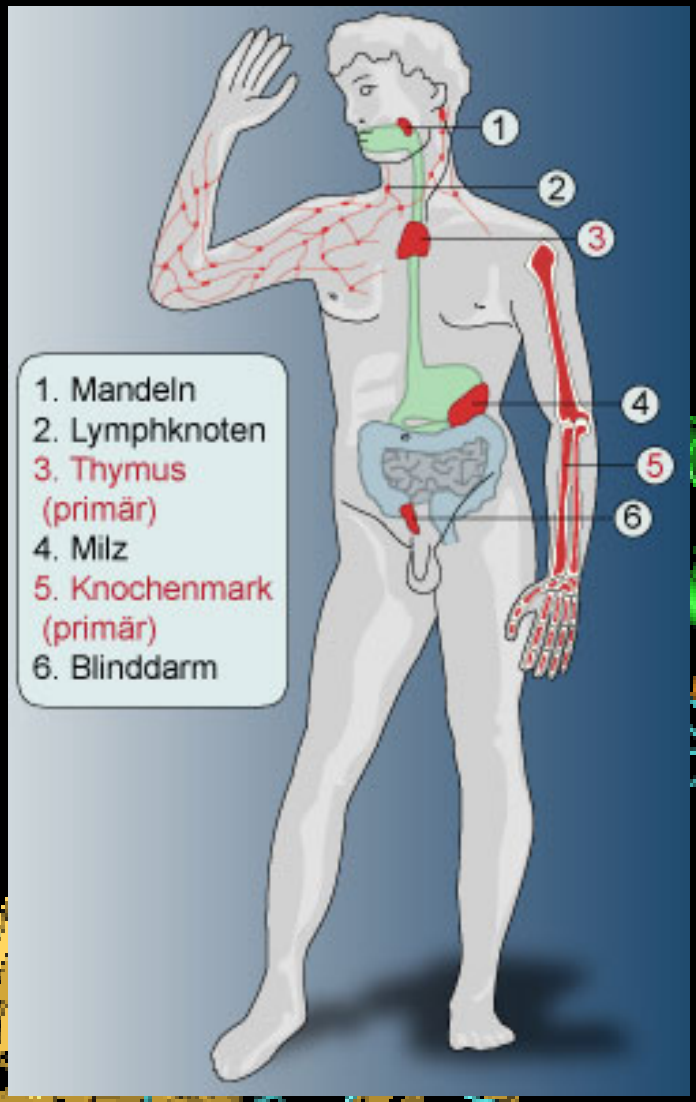
Aufbau des Immunsystems

- Immunsystem des menschlichen Körpers ist außerordentlich komplex, differenziert und in vielen Teilaspekten bis heute noch nicht endgültig verstanden.

- Verschiedene Organe und Zellsysteme sind an der Entstehung der Immunantwort beteiligt:

  <u>zentralen, primären lymphatischen Organe:</u>
  Knochenmark und der Thymus: Diese Organe sind für die Bildung von Lymphozyten zuständig, die dann über das Blut zu den peripheren lymphatischen Organen transportiert werden. Dort wird die erworbene Immunabwehr eingeleitet.

Aufbau des Immunsystems II



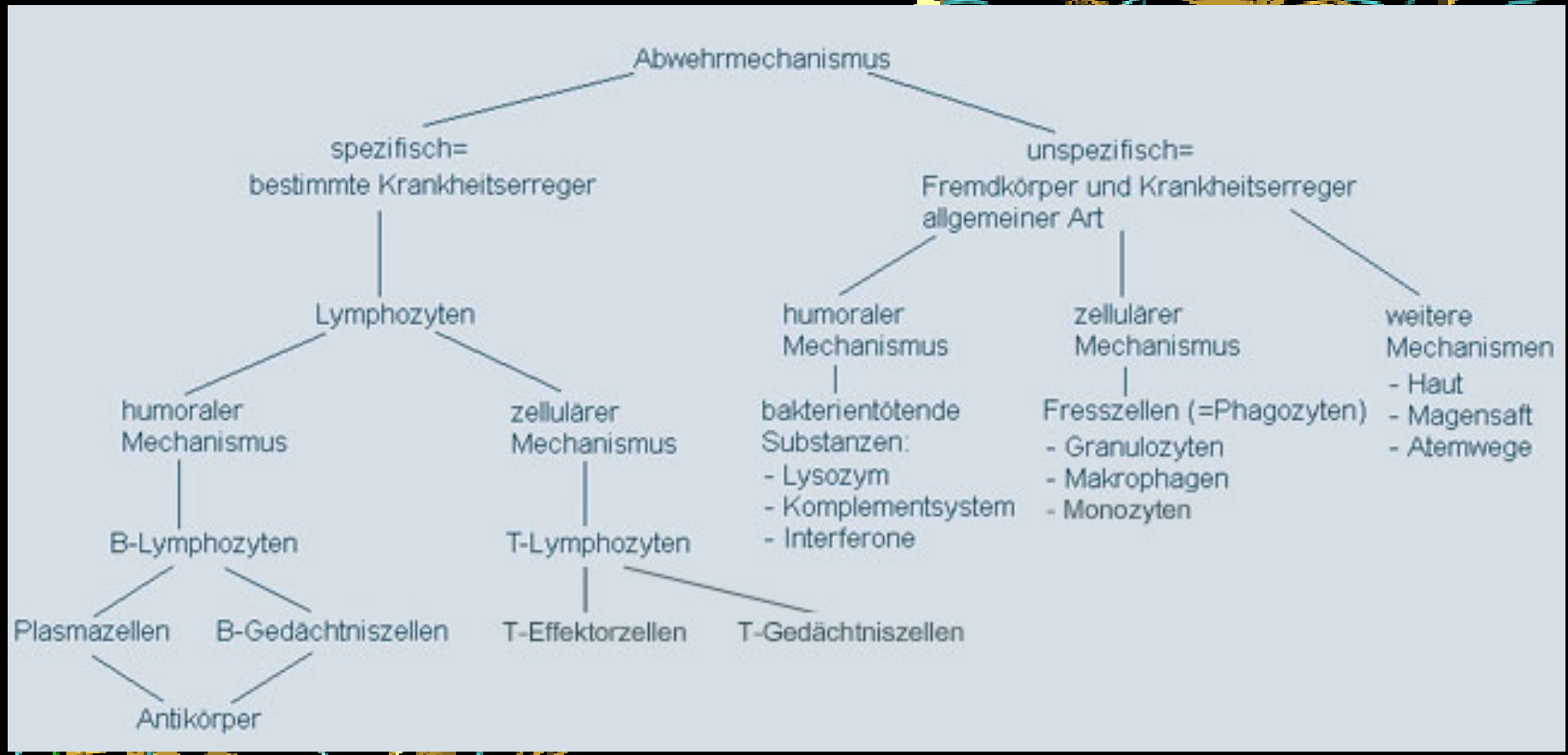
<u>periphere, sekundäre lymphatischen Organe:</u>

Lymphknoten, Milz und die Lymphatischen Gewebe des Magen-Darm-Traktes (Rachenmandeln, Blinddarm u.a.), **der Lunge und anderer Schleimhäute.**

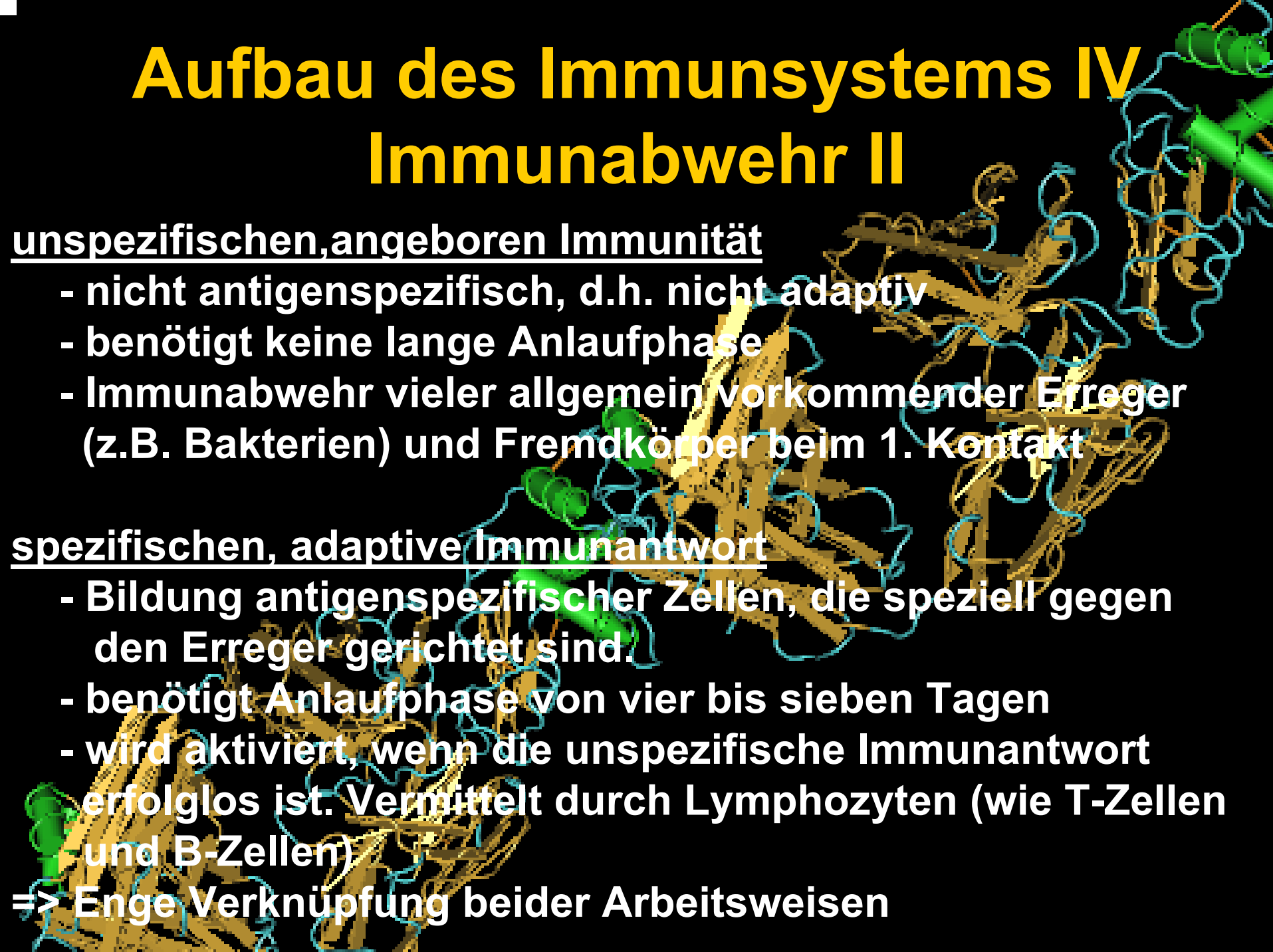
<= Die lymphatischen Organe

Aufbau des Immunsystems III
Immunabwehr

Schematische Darstellung des Immunsystems



Aufbau des Immunsystems IV
Immunabwehr II



unspezifischen,angeboren Immunität

- nicht antigenspezifisch, d.h. nicht adaptiv
- benötigt keine lange Anlaufphase
- Immunabwehr vieler allgemein vorkommender Erreger (z.B. Bakterien) und Fremdkörper beim 1. Kontakt

spezifischen, adaptive Immunantwort

- Bildung antigenspezifischer Zellen, die speziell gegen den Erreger gerichtet sind.
- benötigt Anlaufphase von vier bis sieben Tagen
- wird aktiviert, wenn die unspezifische Immunantwort erfolglos ist. Vermittelt durch Lymphozyten (wie T-Zellen und B-Zellen)

=> Enge Verknüpfung beider Arbeitsweisen

Aufbau des Immunsystems V
Immunabwehr III

- Trotz dieses sich ergänzenden Aufbaus des biologischen Immunsystems, ist nur die adaptive Immunantwort für die informatische Simulation des Immunsystems relevant.

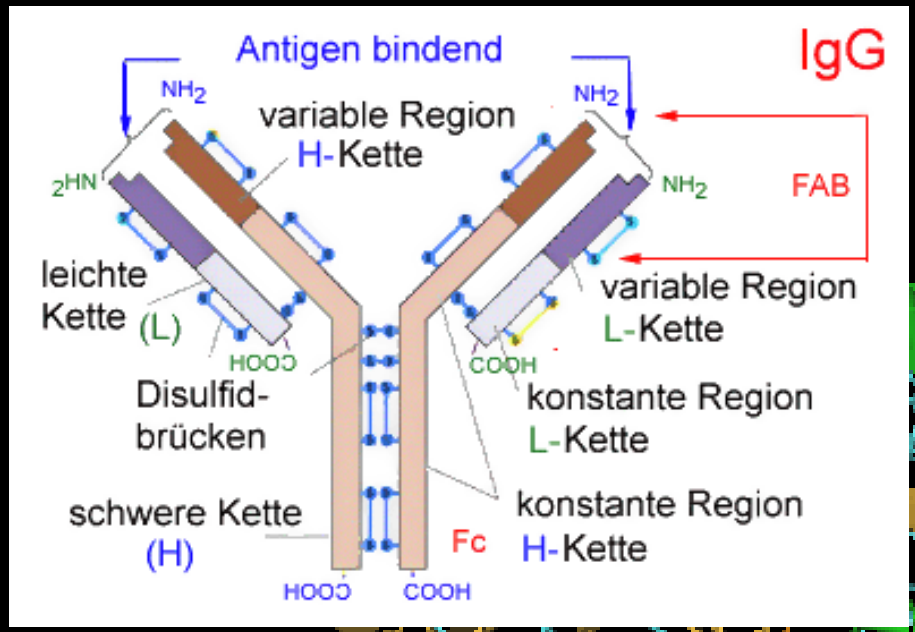
- Daher wird im Folgenden nur adaptive Immunabwehr betrachtet, wobei hier nur die für die Simulation wichtigen Bestandteile dieser Art des Immunsystems, wie Antigene, Antikörper, B-Zellen usw. näher beleuchtet werden.

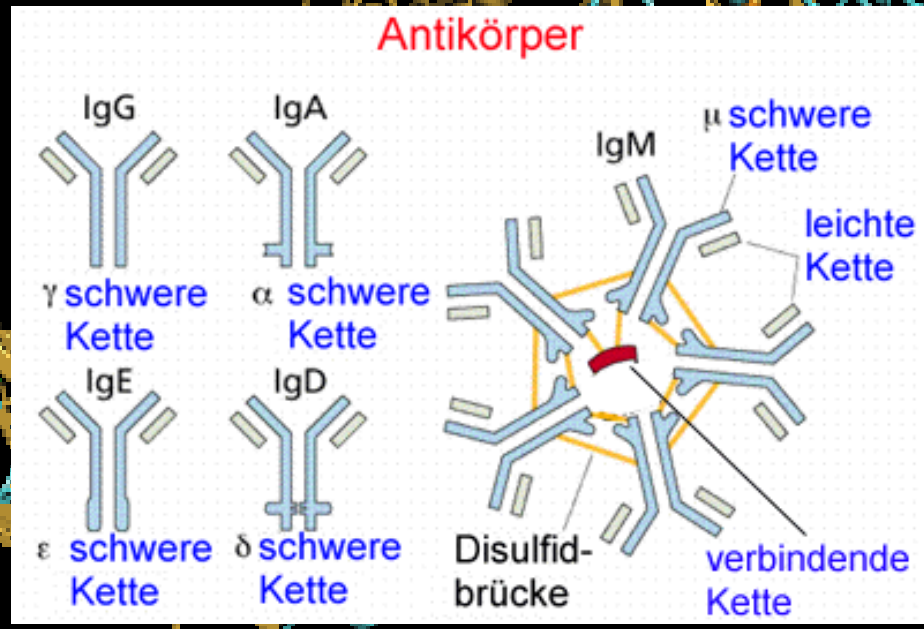
- Für die anderen, nicht dargestellten Teile des biologischen Immunsystems soll in diesem Zusammenhang auf die entsprechende medizinisch-biologische Fachliteratur und diesbezügliche Lehrveranstaltungen verwiesen werden.

Aufbau des Immunsystems VI adaptive Immunantwort

- Ein Hauptproblem der Abwehr ist, daß verschiedene Erregertypen intrazellulär (im Zytoplasma), in Vesikeln (z.B. Viren) und extrazellulär (im Gewebe, auf Epithelzellen) auftreten.
- Zur Bekämpfung extrazellulär auftretender Erreger dienen in erster Linie Antikörper.
  - Ein Antikörpermolekül (=Immunglobulin) besteht aus:
    - 2 leichten (vom Typ κ oder λ) und zwei schweren Polypeptidketten (vom Typ μ, δ, γ, α oder ε) die durch Disulfidbrücken verbunden sind.
    - Je nach Typ der schweren Kette spricht man von IgM, IgG, IgD, IgA oder IgE (Antikörperklassen). Bei IgM sind zusätzlich fünf solcher Grundeinheiten zu einem sehr großen Molekül zusammengefaßt.

Aufbau des Immunsystems VII
adaptive Immunantwort II



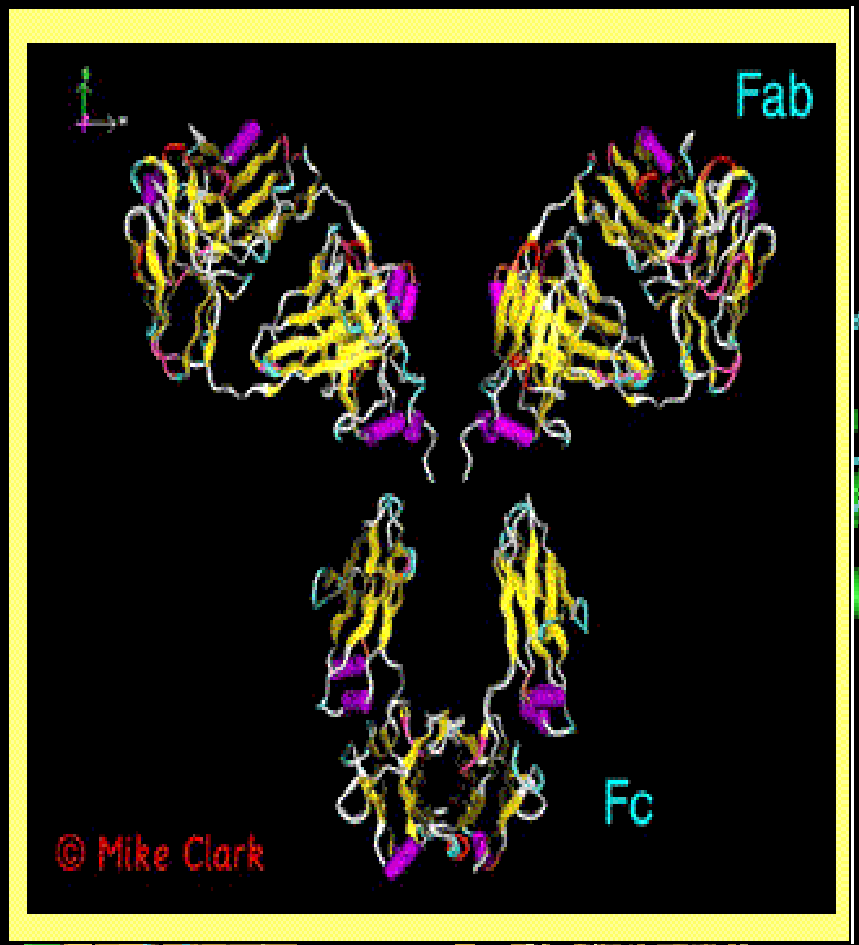


- **Funktionelle Antikörper haben eine variable und eine konstante Region. Während die konstante Region unveränderbar ist, wird die variable Region durch in einem Prozess, das Rearrangement, neu gebildet. Die variable Region dient der Bindung des Antigens.**

Aufbau des Immunsystems VIII adaptive Immunantwort III

- Antikörper (Immunglobuline Ig) werden von Plasmazellen auf den Reiz eines Antigens hin gebildet.
- Die meisten Typen zirkulieren in bestimmten Bereichen des Körpers, einige sind an Lymphozyten gebunden.
- Viele Zellen des Immunsystems haben Rezeptoren, die die den Fc-Teil des Antikörpers binden: so genannte Fc-Rezeptoren. Mit Fc-Rezeptoren erkennen und binden diese Zellen Antigen-Antikörperkomplexe.
- Freie Antikörper werden von den Fc-Rezeptoren nicht gebunden. Eine Ausnahme sind Mastzellen, die über auch freie (nicht an ein Antigen gebundene) IgE an ihre Oberfläche binden.

Aufbau des Immunsystems IX
adaptive Immunantwort IV



- Dargestellt ist das **Immunglobulin G** (IgG). Es ist die häufigste Klasse (80%).
  - Die Antigen-Bindungsstelle liegt in dem verbundenen Teil zwischen der L- und der H-Kette (antigen binding fragment = Fab)
- **IgA** hat ebenfalls eine dimere Struktur. Ca. 10-15% der Immunoglobuline sind vom IgA-Typ.

Aufbau des Immunsystems X adaptive Immunantwort V

- **IgM** kommt zu 5 -10% fast nur im Blut vor und hat eine typische, pentamere Struktur. Als Monomer sitzt er als "Oberflächenantikörper" in der Membran der B-Lymphozyten (sIgM).
- Vom Typ **IgD** werden nur wenig gebildet und seine Funktion ist unklar. Man findet ihn auf der Oberfläche von B-Lymphozyten. Möglicherweise wirkt er als Antigenrezeptor.
- **Die IgE-** Konzentration ist meist sehr gering, da IgEs meist an Rezeptoren auf **basophilen Zellen** und **Mastzellen** gebunden sind.
- Eine reife B-Zelle produziert zunächst IgD und IgM, die an die Membran wandern und als Membranrezeptoren wirken. Bei einer zweiten Stimulierung durch ein Antigen werden dann IgG produziert, die meist eine höhere Antigenaffinität haben, bedingt durch selektive Expansion => Affinitätsreifung

Aufbau des Immunsystems XI adaptive Immunantwort VI

Antigenerkennung:

- Antigen wird alles bezeichnet, was eine adaptive Immunreaktion auslösen kann. Die chemische Zusammensetzung ist von geringer Bedeutung.
- Antikörper erkennen größere, dreidimensionale Oberflächenstrukturen. Oft hat ein Makromolekül mehrere solcher Strukturen, so genannte antigene Determinanten oder Epitope, die unabhängig voneinander Antikörper-Bildung auslösen.
- Umgekehrt können zwei in der Primärstruktur ganz unterschiedliche Moleküle trotzdem vom selben Antikörper erkannt werden, wenn ihre Oberflächenstruktur zufällig sehr ähnlich ist (=Kreuzreaktion).
- T-Lymphozyten erkennen eine andere Art von Epitopen : lineare Peptide von 8 bis 20 Aminosäuren => ca. 10^{16} (10^{18}) Antikörper

Aufbau des Immunsystems XII
adaptive Immunantwort VII

Antikörpervielfalt generiert durch:

- die kombinatorische Vielfalt innerhalb der Antikörper bildenden Polypeptidketten (somatische Rekombination).
- die kombinatorische Vielfalt durch Kombination von schweren und leichten Ketten im fertigen Antikörper. Es ergeben sich dadurch bereits $3{,}5 \times 10^6$ verschiedene Antikörpermoleküle
- die junktionale Variabilität der Strukturesegmente für die variable Region der schweren Kette
- die somatische Hypermutation (Punktmutationen in den Kodons)

Aufbau des Immunsystems XIII adaptive Immunantwort VIII

Antikörperselektion:

- Zellklone, Antikörper produzieren, die ubiquitäre körpereigene Antigene erkennen, werden in einem frühen Entwicklungs-stadium abgetötet (klonale Deletion) oder in ein Stadium gebracht, in dem sie zwar bestehen bleiben, aber nicht mehr reagieren können (klonale Anergie). =>Keine perfekten Schutzmechanismen
- klonale Selektion: das Antigen sucht sich die B-Zelle mit dem passenden Antikörper auf der Oberfläche aus und regt diese zur Bildung von Tochterzellen an, die alle denselben "nützlichen" Antikörper bilden.
  - Aber wegen der möglichen Bildung von Autoantikörpern ist die T-Zell-Hilfe als zweite Bedingung notwendig, um die um die Antikörperproduktion in Gang zu bringen.

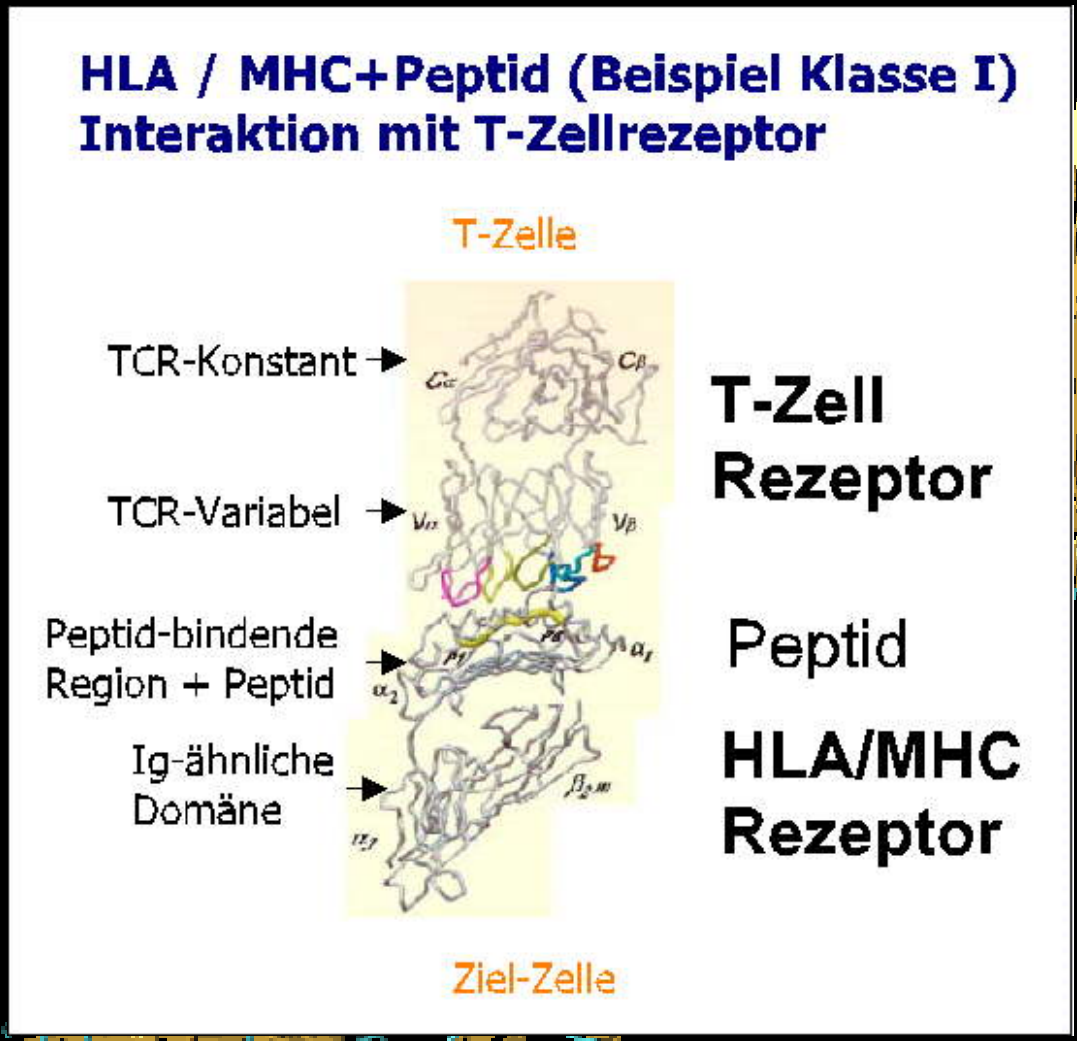
Aufbau des Immunsystems XIV adaptive Immunantwort IX

- **Lymphozyten:**
  - sind die kleinsten weißen Blutkörperchen.
  - Ihr Anteil an der Gesamtmenge der weißen Blutkörperchen im Blut beträgt etwa ein Viertel.
  - Allerdings befinden sich 98 % der Lymphozyten nicht im Blut, sondern in den lymphatischen Organen (Lymphknoten, Lymphbahnen, Milz) und im Knochenmark. Von dort aus wird ständig ein kleiner Teil der Zellen ins Blut abgegeben.
  - Lebensdauer der Lymphozyten: zwischen zehn Tagen und mehreren Jahren.
  - entwickeln sich zunächst im Knochenmark und im Thymus, d.h. den primären Organen des Immunsystems, und besiedeln von dort aus die sekundären Immunorgane wie Lymphgewebe und Milz.

Aufbau des Immunsystems XV adaptive Immunantwort X

- **T-Lymphozyten:**
  - im Thymus gebildet und darauf geprägt, zwischen körper-eigenen und körperfremden Strukturen zu unterscheiden.
  - Bei Fremdkörperkontakt Entwicklung zu T-Effektorzellen oder T-Gedächtniszellen (Erkennung des gleichen Fremd-körpers nach Jahren)
  - Die T-Effektorzellen lassen sich in zwei Gruppen einteilen:
    - T-Helferzellen (TH):
      - Typ1(TH1-Zellen): Hauptfunktion: Hilfe der Makrophagen beim Abtöten intrazellulärer Erreger.
      - Typ2(TH2-Zellen): leisten Hilfe zur Antikörperproduktion (B-Zellen).
    - Zytotoxische T-Zellen, die durch Lyse (=Auflösung) (Virus-)infizierte Zellen töten können.

Aufbau des Immunsystems XVI
adaptive Immunantwort XI



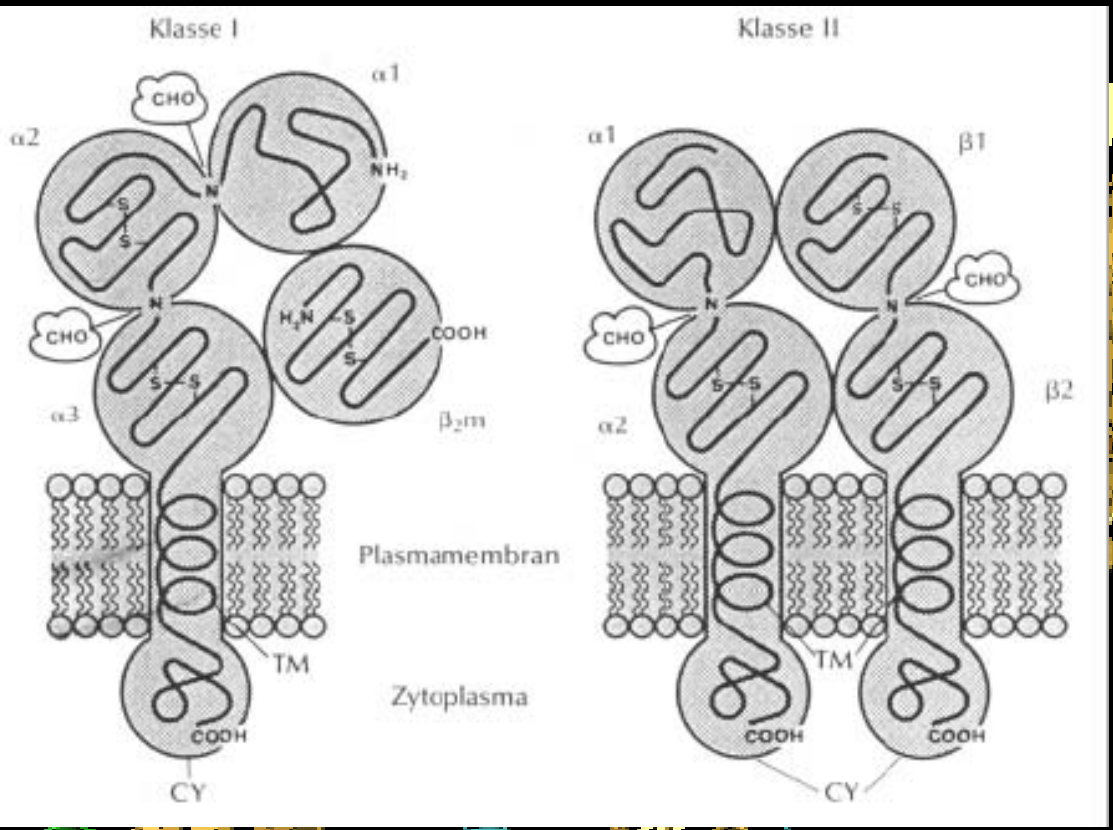
- besitzen auf ihrer Oberfläche einen T-Zellrezeptor, der kurze auf MHC-Molekülen präsentierte Antigen-Peptide erkennt, die in den MHC-Spalt eingebaut sind.
- ähnelt grob dem Fab-Fragment eines Antikörpers: er besteht aus zwei Ketten (α:β bzw. γ:δ), und hat an der "Spitze" eine variable Region

Aufbau des Immunsystems XVII adaptive Immunantwort XII

Major histocompatibility complex, MHC)

- beim Menschen human leucocyte antigen“ (HLA)- System genannt (für jeden Locus gibt es zahlreiche Varianten.)
- **3 MHC-Klassen werden kodiert:**
  - MHC Klasse I-Moleküle,von fast allen kernhaltigen Zellen exprimiert agieren mit zytotoxischen T-Lymphozyten und regulieren sie MHC-konzentrationsabhängig
  - MHC-Klasse-II-Moleküle,nur auf B-Lymphozyten, Makrophagen, dendritischen Zellen und Endothelzellen sowie auf manchen T-Lymphozyten exprimiert, agieren mit TH-Zellen und können längere Peptide binden als Klasse I MHC-Moleküle.
  - MHC Klasse III-Moleküle kodieren für verschiedene Komplementkomponenten

Aufbau des Immunsystems XVIII
adaptive Immunantwort XIII



- auf der T-Zelloberfläche positioniert (Unterscheidung zwischen körpereigenen und fremden Antigenen durch T-Lymphozyten)

<=Heterodimer, unterteilt in eine extrazelluläre, eine transmembrane und eine zytoplasmatische Region.

Aufbau des Immunsystems XIX adaptive Immunantwort XIV

- Peptidfragmente von extrazellulären Proteinen binden normalerweise an Klasse II MHC-Moleküle, endogen synthetisierte Proteine an Klasse I MHC-Moleküle.

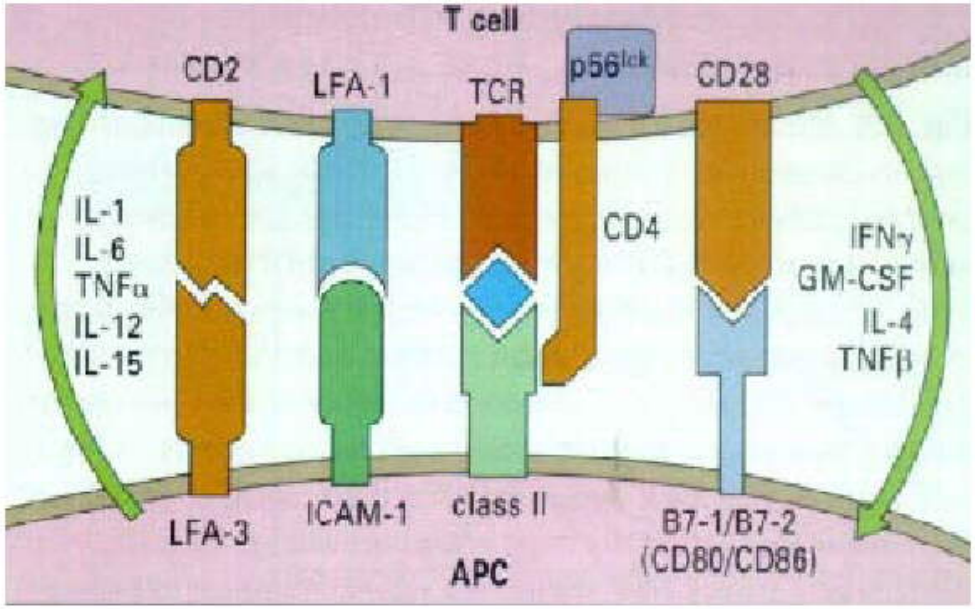
- Notwendige Antigenpräsentation mit MHC-II-Molekülen an der Oberfläche von Antigen-präsentierenden Zellen (APC) für die Induktion verschiedener T-Lymphozyten. Denn T- Zellen können Peptid-Antigene nur als MHC-Komplex erkennen.

- Zytotoxische T-Zellen (CD8-Oberflächenmarker positiv) sind MHC I restringiert, TH-Zellen (CD4-positiv) MHC II restringiert.

- Nur die extrazellulären Domänen von CD4 und CD8 binden an Klasse II- *und* Klasse I-MHC und agieren so als Adhäsionsmoleküle, die den Kontakt zwischen der T-Zelle und den APCs verstärken.

Aufbau des Immunsystems XX
adaptive Immunantwort XV

MHC-II -CD4 T-Helfer Interaktion Kostimulation(naive T-Zellen)



1. T-Zellrezeptor	MHC-II Peptid-Komplex	Aktivierung
2. CD4	MHC Klasse-II	Ko-Aktivierung
3. CD28	B7	Kostimulation
4. CD40 Ligand	CD40	Kostimulation
5. Andere(CD2/Ligand)	Andere (LFA-3/Adhesine)	Adhäsion
6. Zytokine	Zytokinrezeptoren	Aktivierung

- CD4-T-Zellen verlieren ihre Reaktionsbereitschaft, wenn sie Peptidantigene erkennen, die von anderen T-Zellen, die MHC II-Moleküle exprimieren, präsentiert werden.

Aufbau des Immunsystems XXI adaptive Immunantwort XVI

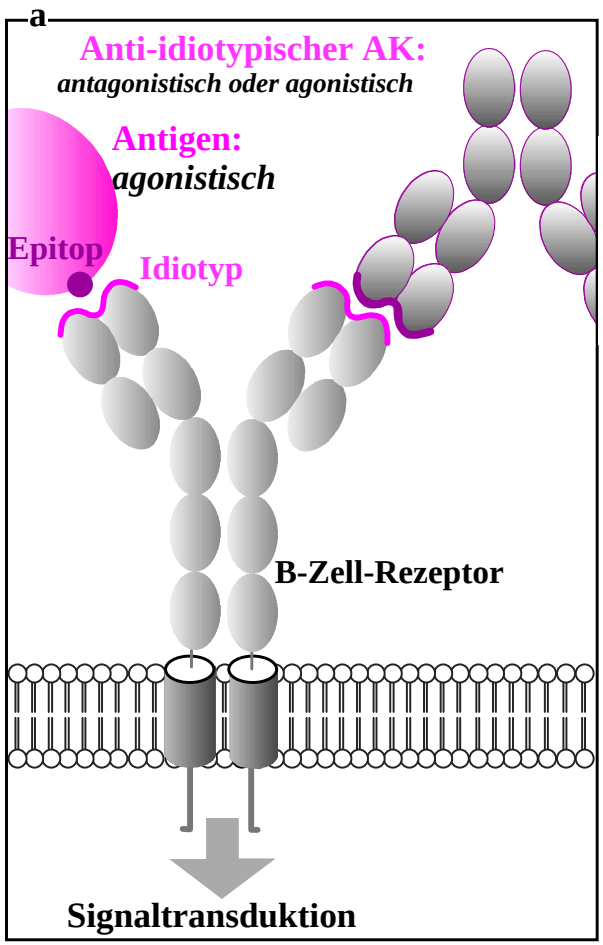
Selektion nützlicher T-Zellen

- Die Vorstufen fertiger T-Zellen im Thymus, die Thymozyten reifen in einem zweistufigen Prozess. Ziel: nur nützliche T-Zellen verlassen den Thymus:
  - **Positive Selektion**: Thymozyten besitzen einen einzigartigen T-Zellrezeptor. Passt die Rezeptorstruktur nicht mit den eigenen MHC-Molekülen zusammen, ist er nutzlos und tötet sich selbst. Denn ein erfolgreiches Andocken des T-Zell-Rezeptors an das MHC-Molekül des Thymusepithels entsteht Überlebenssignal für den Thymozyten.
  - **Negative Selektion**: Passt der T-Zellrezeptor des Thymozyten perfekt auf eine Kombination von eigenem MHC mit darauf präsentiertem Selbst-Peptid, dann ist er autoreaktiv und damit gefährlich. Aber durch die induzierte sehr starke Bindung, die zu einem qualitativ anderen Signal führt, wird im Thymozyten Apoptose eingeleitet.

Aufbau des Immunsystems XXII
adaptive Immunantwort XVII

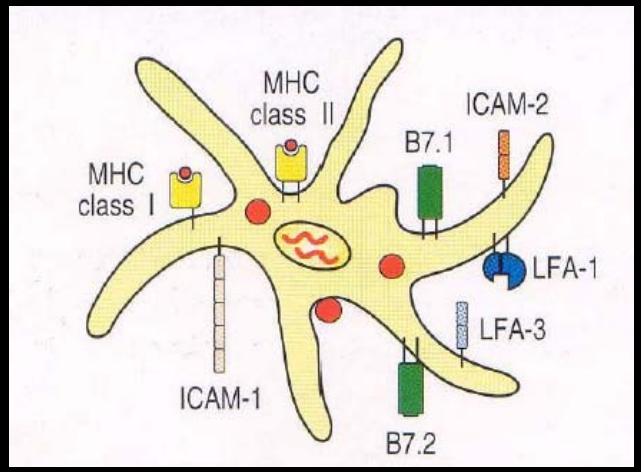
- **<u>B-Lymphozyten:</u>**
  - reifen im Knochenmark und machen etwa 15 % aller Lymphozyten im Blut aus. Es gibt ca. 10^{10} B-Zellen und Antikörperrezeptoren
  - Bei Fremdkörperkontakt entwickelt sich ein Teil der B-Lymphozyten zu Plasmazellen, die Antikörper gegen diesen Fremdkörper bilden. Plasmazellen leben etwa 2-3 Tage.
    - zunächst Erzeugung von IgM; durch *class switch* in einem Teil der Zellen wird später IgG bzw. IgA produziert.
  - Der anderen Teil der B-Lymphozyten wird nach Kontakt im Lymphknoten mit einem Fremdkörper zu langlebigen B-Gedächtniszellen, die noch Jahre später, auch wenn der Körper nicht mehr diesem Fremdkörper ausgesetzt ist, die gleichen Antikörper bilden (→ <u>schnellere Reaktivierung</u>)

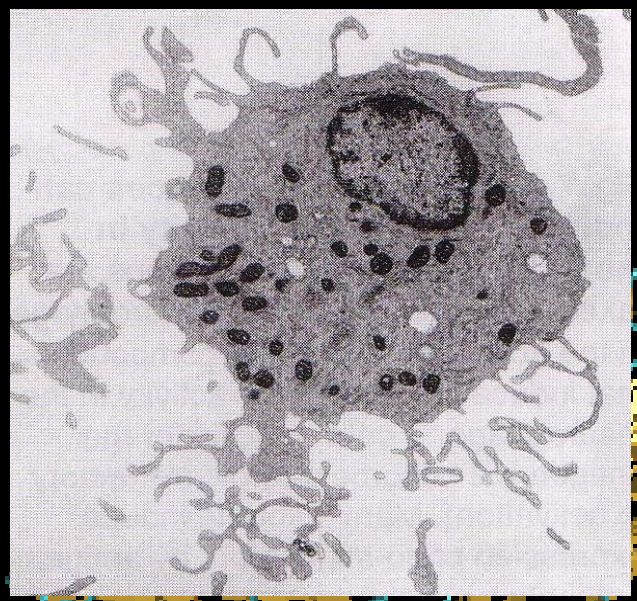
Aufbau des Immunsystems XXIII
adaptive Immunantwort XVIII



- Besitzen einen Zellrezeptor
- Hat eine B-Zelle einen passenden Rezeptor für ein Antigen, nimmt sie das Pathogen auf, verarbeitet das Antigen und präsentiert es auf MHC-II-Molekülen.
- Erkennt eine TH2-Zelle das auf dem B-Zell-MHC-II-Komplex präsentierte Antigen-Peptid für das sie aktiviert ist, und reagiert, wird die B-Zelle aktiviert, d.h sie beginnt ihrerseits zu proliferieren.
- Die T-Zelle proliferiert auch

Aufbau des Immunsystems XXIV adaptive Immunantwort XIX

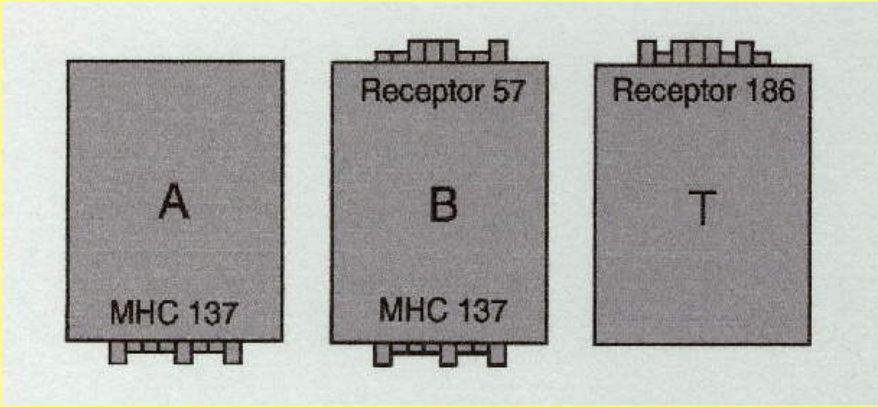




Antigenpräsentierende Zellen:

- Dendritische Zellen ("professionelle APC")
  - besonders reich an unspezifischen Rezeptoren zur Antigenaufnahme
  - Rezeptoren/Kostimulatoren zur Präsentation bakterieller und viraler Antigene für die Stimulation von B-Zellen (über B-Zellrezeptor) und Aktivierung/Stimulation zytotoxischer T-Zellen bzw. TH2-Zellen.
- Makrophagen, Langerhans-Zellen, interdigierende Follikelzellen (Thymus), B-Zellen

Simulation des Immunsystems
Modell IMMSIM



Schema einer Antigenpäsentierenden Zell (APC), einer B-Zelle und einer T-Zelle mit einem 8-bit Rezeptor. Die integer Namen des 8-bit Segments sind die Dezimalwerte der binären Strings. MHC steht für major histocompatibility complex.

Für IMMSIM (immune simulation) steht die Gewinnung und Prozessierung von Antigenen und Beeinflussung verschiedener Zellpopulationen im Blickpunkt. Damit können Aspekte der klonalen Selektion (Mutation, Affinitätsreifung), Thymus und Toleranz betrachtet werden.

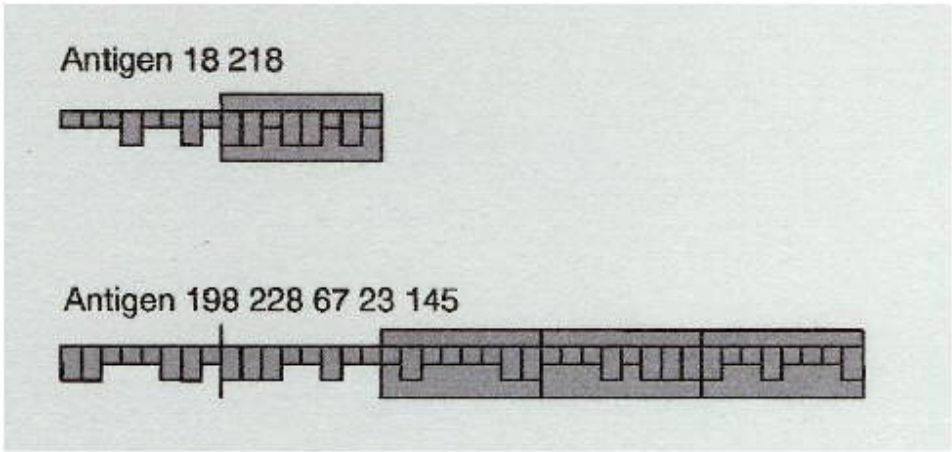
Bestandteile des Modells:

- B-Zellen, T-Zellen, APCs, Antikörper, Antigene und Antigen-Antikörper-Komplexe

Simulation des Immunsystems II
Modell IMMSIM II

- B-Zellen:
  - Jede besitzt einen exponierten Antigen-spezifischen Rezeptor als n-Bit String. Antikörpergewinnung und Prozessierung erfolgt durch B-Zellen
  - MHC als n-Bit-String dargestellt
- T-Zellen:
  - Jede besitzt einen n-Bit String (der eigene Rezeptor)
  - bindet an die MHC der B-Zellen und APCs
- APC:
  - MHC besitzt einen n-Bit-String
- Antigen:
  - Set von Bitstrings, von denen einige Epitope, andere Peptide repräsentieren

Simulation des Immunsystems III
Modell IMMSIM III



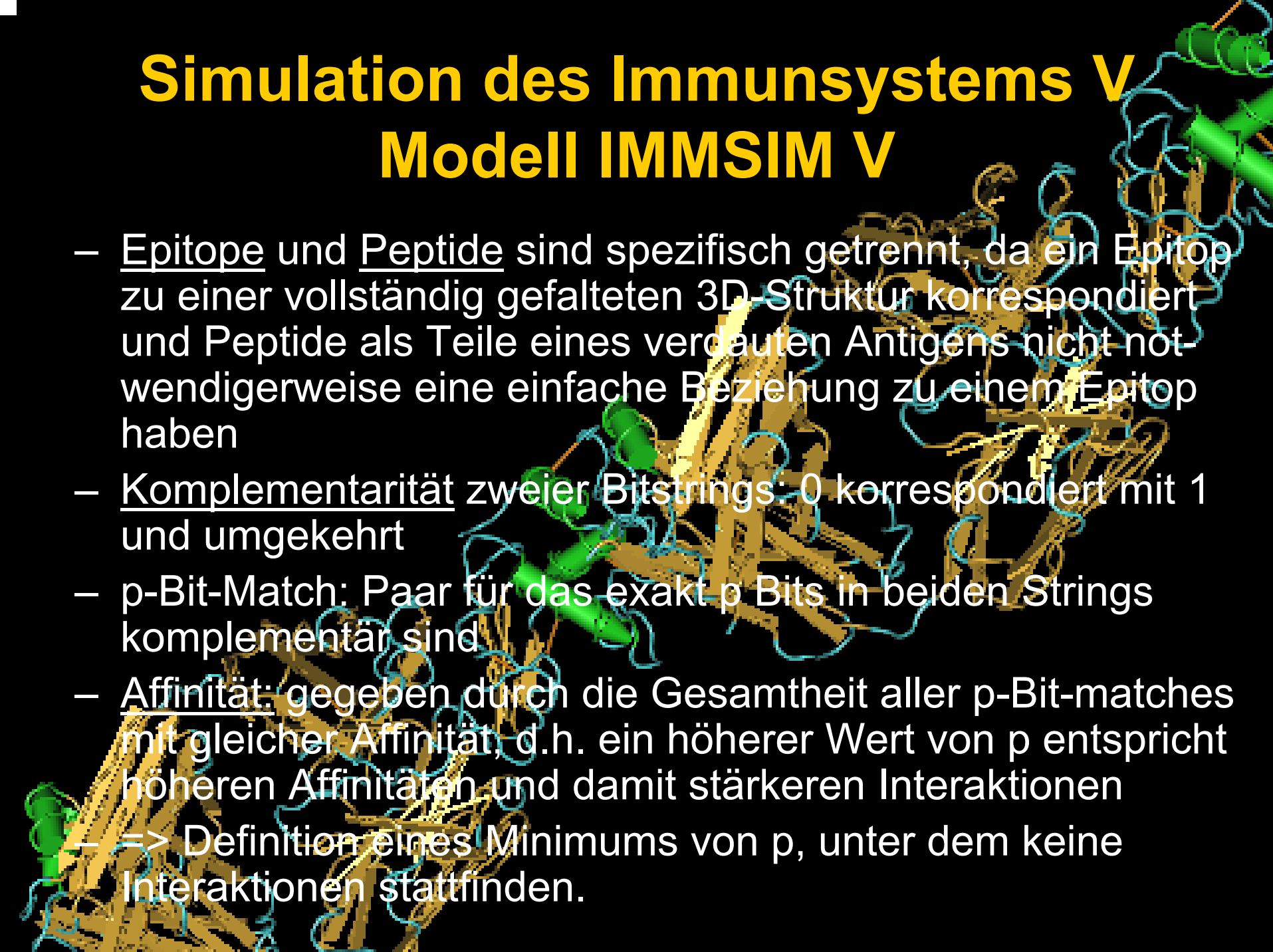
Beispiele für Antigene eines 8-Bit-Systems
Peptide sind eingeschachtelt dargestellt.
Epitope sind nicht eingeschachtelt. Die
ganzen Zahlen bezeichnen das jeweilige
Peptid oder Epitop als Teil des Antigens

- Antikörper:
  - besitzen den gleichen Rezeptor wie B-Zelle, von der sie abstammten
  - sie können auch andere Rezeptoren oder Peptide haben.
- Antigen-Antikörper-Komplex:
  - Bindung des Antikörpers an das Antigen über Epitope

Simulation des Immunsystems IV
Modell IMMSIM IV

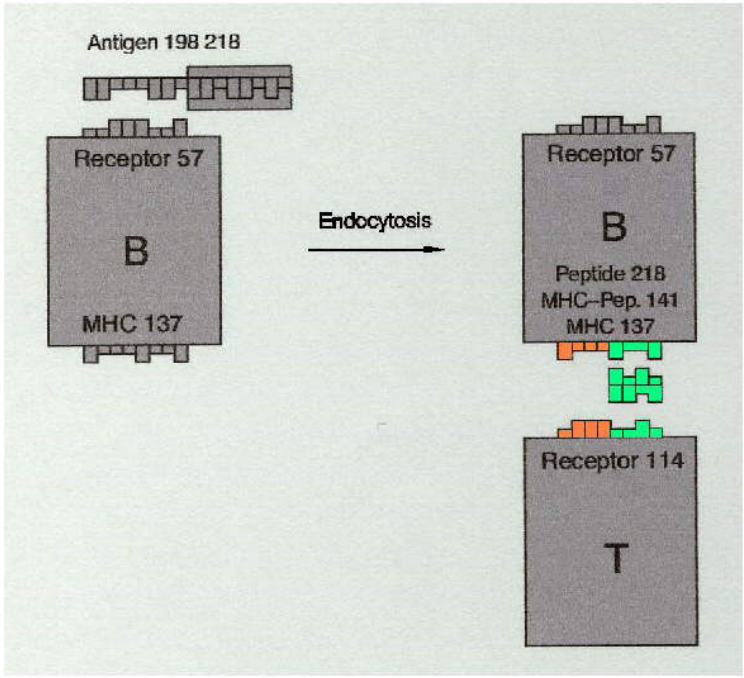
- Rezeptorpräsentation:
  - Bit-String (0s und 1s) entspricht einem Oberflächenrezeptor
  - Je grösser das Repertoire der verschiedenen Antikörper ist, um so grösser ist die Zahl der verwendeten Bits.
  - Ein n-bit-String ergibt 2^n Spezifikationen, d.h. z.B. für einen 8-Bitstring: **2^8 =256 Spezifikationen**
- ***Definition und Regeln:***
  - Epitop: Teil eines Antigens, das durch einen B-Zellrezeptor erkannt wird
  - Peptid: Teil eines Antigens, das durch ein MHC-Molekül gebunden werden kann und durch eine geeignete T-Zelle erkannt wird.

Simulation des Immunsystems V
Modell IMMSIM V



- Epitope und Peptide sind spezifisch getrennt, da ein Epitop zu einer vollständig gefalteten 3D-Struktur korrespondiert und Peptide als Teile eines verdauten Antigens nicht notwendigerweise eine einfache Beziehung zu einem Epitop haben
- Komplementarität zweier Bitstrings: 0 korrespondiert mit 1 und umgekehrt
- p-Bit-Match: Paar für das exakt p Bits in beiden Strings komplementär sind
- Affinität: gegeben durch die Gesamtheit aller p-Bit-matches mit gleicher Affinität, d.h. ein höherer Wert von p entspricht höheren Affinitäten und damit stärkeren Interaktionen
- => Definition eines Minimums von p, unter dem keine Interaktionen stattfinden.

Simulation des Immunsystems VI
Modell IMMSIM VI



Schritte in der Erkennung und Prozessierung eines Antigens durch B- und T-Zellen
Die MHC-T-Zellrezeptorbindungsstelle ist orange, die Peptid-T-Zellbindung ist grün dargestellt

B-Zell-Aktivierung

- Probabilistische Bindung der B-Zelle an Antigenepitope, wobei diese von der Zahl der gematchten Bits abhängt (je grösser die Zahl der Bit-matches, um so grösser die Interaktions-Wahrscheinlichkeit).
- Nach Aufnahme des gebundenen Antigens, werden die antigenen Peptide an MHC-Moleküle gebunden und auf der B-Zelloberfläche exponiert (1.Signal der Aktivierung).
- Dann erfolgt die T-Zell-Bindung (2.Aktivierungssignal)

Simulation des Immunsystems VII
Modell IMMSIM VII

- ***Simulation der MHC-Peptid-Bindung***

  MHC-Bitstring wird in der Hälfte geteilt:
  - Linke Hälfte: MHC-Teil, der den T-Zellrezeptor bindet.
  - Rechte Hälfte: MHC-Teil, der die Grube repräsentiert, in welcher das Peptid bindet.
  - Wenn es nur ein MHC und ein Peptid gibt, kann entweder die linke oder die rechte Hälfte des Peptids binden.
  - Bei Bindung des MHC mit einer Hälfte des Peptids wird die andere Peptidhälfte den T-Zellen präsentiert
  - Der T-Zellenrezeptor bindet den MHC-Peptid-Komplex mit den gleichen Regeln wie ein B-Zellrezeptor an das antigenische Epitop.
  - Bei erfolgreicher T-Zellbindung, T- und B-Zelle teilen sich und etablieren Klone ihres eigenen Typs:
  - T-Zellen teilen sich dreimal => $2^3 = 8$ Zellen

Simulation des Immunsystems VIII
Modell IMMSIM VIII

- B-Zellen teilen sich viermal = 16 Zellen:
  - die eine Hälfte, also 8 Zellen werden Gedächtniszellen (wie die T-Zellen)
  - die andere Hälfte, ebenfalls 8 Zellen, werden Plasmazellen und produzieren Antikörper
  - Sich teilende B-Zellen durchlaufen auch Hypermutationen durch zufälligen Einbau von 1-Bit-Austauschen in ihren spezifischen Bitstringrezeptor

- **Antigenpräsentierende Zelle (APC):**
  - Nichtspezifische Bindung und Prozessierung von Antigenen
  - Prozessieren Peptide und präsentieren sie
  - Nur T-Zellen teilen sich, wenn eine T-Zellbindung an den MHC-Peptid-Komplex auf dem APC erfolgt.

Simulation des Immunsystems IX
Modell IMMSIM IX

Beispiel individuelle eigenschaftsbezogener Zellentwicklung während der Simulation:

- B-Zelle kann eine gerade im Knochenmark gebildet (naive) Zelle sein:
  - Sie bindet ein Antigen und präsentiert das antigenisch Peptid. B-Zelle wird durch dieses Antigen charakterisiert und durch den MHC-Peptidkomplex vorgestellt.
  - Dann kann sie durch eine T-Zelle stimuliert werden
    - => Phase der Teilung und Differenzierung in Gedächtnis- und Plasmazelle
    - => Plasmazellen beginnen, Antikörper zu produzieren
    - => Gedächtniszellen können den Prozess neu starten
- **Antigen-Antikörperkomplexe werden wie verklumpte Moleküle** behandelt, charakterisiert nur durch ihre Epitope und Peptide. Sie haben keinen inneren Zustand.

Simulation des Immunsystems X
Modell IMMSIM X

- Naive B- und T-Zellen werden zufällig mit kompletter Diversität (2^8 = 256 Typen für 8 Bits) gewählt.

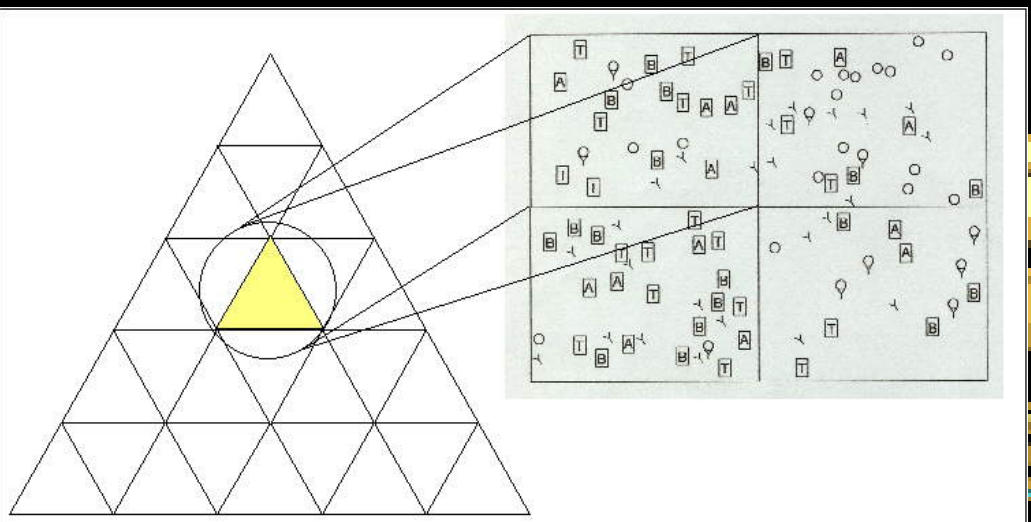
- **Veränderung der T-Zelldiversifikation durch thymische Interaktionen während der Modellthymuspassage:**
  - Negative Selektion: T-Zellen werden dem MHC-Eigenpeptid-komplex u. U. viele Male ausgesetzt und im Verhältnis zur Bindungsstärke eliminiert. Je grösser die Anzahl der Tests ist, um so effizienter ist die Eliminierung der eigenreaktiven T-Zellen.
  - Positive Selektion, d.h. nur potentiell MHC bindungsfähige T-Zellen verlassen den Thymus (z.B. bei 8 Bits 7 matchende Bits => mindestens 3-Bit für MHC-Peptid-Bindung)

- Die Zahl der überlebenden T-Zellen ist von der Zahl und dem Charakter der verfügbaren MHCs und Eigenpeptide abhängig

Simulation des Immunsystems XI
Modell IMMSIM XI

- **Verwendung eines modifizierten zellulären Automaten:**
  - Dynamisches System, in dem Raum und Zeit diskret sind
  - Jede Stelle nimmt einen begrenztes Set möglicher Werte an.
  - Entwicklung der Werte an jeder Stelle nach den gleichen deterministischen Regeln.
  - Die Regeln für die Entwicklung einer Stelle hängt von einem lokalen Nachbarn, der diese Stelle umgibt, ab.
- Im IMMSIM-Modell:
  - Ersatz von deterministisch durch probabilistisch
  - Wählt aus der lokalen Nachbarschaft eine Stelle aus, die er selbst wird. Den Bestandteilen wird erlaubt, sich von Stelle zu Stelle zu bewegen
  - Das Gitter ist dreieckig, gedacht als kleiner Teil eines Körpers. Jede Stelle kann mit einer Zahl verschiedener Bestandteile einiger Typen bevölkert werden

Simulation des Immunsystems XII
Implementierung IMMSIM



Zeichnung von 4 Stellen in dem Automaten mit einem dreieckigen Gitter. Die jeweilige korrespondierende Gitterstelle ist farbig hervorgehoben. Gezeigt ist die rechteckige Anordnung der normalerweise dreieckigen Gitterstellen. A,B und T bezeichnen APCs, B-Zellen und T-Zellen. Die Kreise sind Antigene und die Ys Antikörper.
Die Kreise, die an Antikörper gebunden sind, stellen Antigen-Antikörper-Komplexe dar. Zu irgendeinem Zeitschritt können nur die Bestandteile in der gleichen Stelle interagieren.

- ***Start der Simulation:***
  - Zufällige Bevölkerung der Stellen mit der gewünschten Zahl von B-Zellen, *T*-Zellen und APCs.
  - Wahl eines Verzeichnises von Antigeninjektionen und die Simulation beginnt

Simulation des Immunsystems XIII
Implementierung IMMSIM II

- **Interaktionsregeln :**
  - Alle möglichen Interaktionen werden betrachtet
  - Wenn ein Bestandteil fähig ist, mehr als eine erfolgreiche Interaktion zu haben, wird die aktuelle zufällig bestimmt. Nachdem alle Interaktionen entschieden sind, folgt die Geburt neuer Zellen.
- Zusätzlich zum klonalen Wachstum werden neue, naive Zellen produziert.
- Zellen wird auch erlaubt, nach einer gegebenen Halbwertszeit zu sterben.
- Letztlich alle Wesen können in die sie umgebenden Stellen diffundieren = 1 Zeitschritt
- Wiederholungen dieses Vorgangs wie zuvor bestimmt

Simulation des Immunsystems XIV
Implementierung IMMSIM III

Darstellung der Simulationsergebnisse einer Immunisierung:

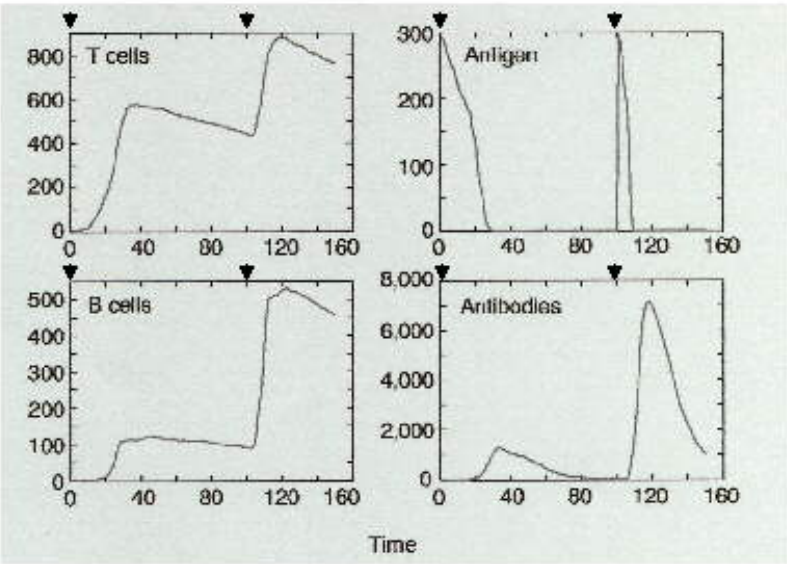
t = 0: (langsame Antwort)

- Entwicklungszeit der Klone der antwortenden Zellen zur Populationsvergrösserung
- Produktion einer geringen Antikörperkonzentration
- Zeit zur Antigenentfernung aus dem System notwendig.

t = 100 (schnelle Antwort)

- B- und T-Zellenkonzentration ist hoch
- => Viel mehr Antikörper werden produziert
- => Antigene werden schnell eliminiert.

Simulation des Immunsystems XV
Implementierung IMMSIM IV
Das Programm



Entwicklung der B- Zell, T-Zell, Antigen- und Antikörperpopulation eines Systems, das mit Antigen zu den Zeitschritten 0 und 100 injiziert wurde (siehe Pfeile).

IMMSIM Das Programm

- vereinfachte Version enthält 3 Bestandteile (je 1 Datenstruktur, die wenigstens auf 2 Dimensionen zur Gitterpositionsbeschreibung basieren):
  - **B-Zellen, Antikörper und Antigene**
  - **Interaktionsmöglichkeiten:**
    - B-Zellen binden Antigene
    - Antikörper binden Antigene

Simulation des Immunsystems XVI
IMMSIM Das Programm II

```
/* B cell structure */

enum bType {NAIVE,PLASMA,MEMORY,ANY}; /* Types of B cells */

struct BCELL {

  /* NOTE:  The number of matching bits with antigen (match) is pre-calculated when
        the B cell is created and must be changed whenever sIg is altered */

  struct BCELL *  next;     /* Linked list pointers */
  struct BCELL *  prev;

  enum bType    type;     /* Naive, Memory, Plasma */
  short     divide;     /* Number of divisions left to undergo */
  unsigned long sIg;      /* Surface immunoglobulin bitstring receptor */
  unsigned    match;      /* Number of matching bits with antigen, based on sIg (pre-calculated) */
};
....

....
unsigned    abLattice[2][_bits + 1][_xDim][_yDim];        /* Antibody -- second index gives number of
matching bits with antigen */
unsigned    agLattice[2][_xDim][_yDim];             /* Antigen */
struct BCELL *  bLattice[2][_xDim][_yDim];              /* B cell */
```

Diffusion

↓

Das Array für die **Antigen-Speicherung** hat 2 Dimensionen, wobei jedes Arrayelement ein Integer ist und der Antigenkonzentration an dem Gitterpunkt, festgelegt die Arrayindizes, entspricht.

Simulation des Immunsystems XVII
IMMSIM Das Programm III

- Strukturelle Annahme: Es gibt nur ein einziges Antigen, dass nicht mutiert. Dies ist für die Strukturausdehnung auf das Modell eines mutierenden Virus (HIV) wichtig.
- Das Antikörper-Array beinhaltet eine Extra-Dimension im Bereich von 0 bis n, wobei n die Bit-Länge ist.
  - Die Arrayelemente sind Integers, die die Zahl der Antikörper an der korrespondierenden Gitterstelle und die Zahl der matchenden Bits mit Bezug auf das Antigen, das durch den Index des Extraarrays gegeben ist, darstellen.
  - Nur die Zahl der matchenden Bits mit dem Antigen ist notwendig, da sie die Stärke der Interaktion bestimmt.
  - Deshalb kann der Speicherbedarf und die Rechenzeit durch einfaches Zusammenfassen der Antikörper mit der gleichen Zahl an matchenden Bits geschont werden.

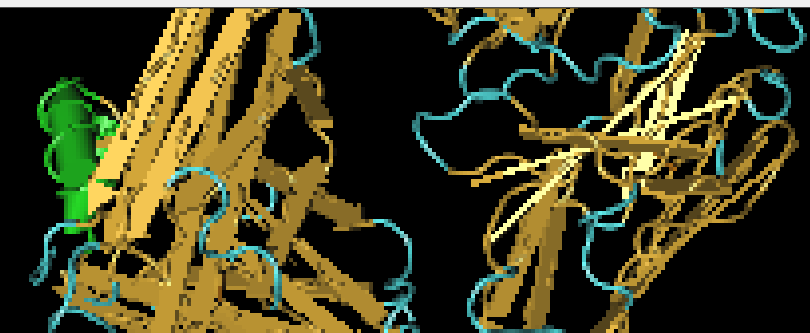
Simulation des Immunsystems XVIII
IMMSIM Das Programm IV

- **B-Zellen**
  - Haben wie die Antigene 2 Dimensionen
  - Jedoch die Arrayelemente sind in diesem Fall verbundene Listen, die die individuellen B-Zellen enthalten, die sich aktuell an der Stelle befinden.
  - Jede B-Zelle wird individuell gespeichert
  - Die B-Zell-Struktur definiert die Information, die mit jeder B-Zelle gespeichert wird:
    - Bit-String-Antikörperrezeptor,
    - B-Zelltyp und
    - B-Zellteilung

- Außerdem besitzt jedes dieser Arrays eine zusätzliche Dimension, die für die Art des Diffusionsprozesses steht.

Simulation des Immunsystems XIX
IMMSIM Das Programm V

```
/* Main simulation loop */

  for (simTime = 0; simTime < 200; simTime++) {

    /* Simulate processes */

    birth();        /* B cell proliferation */
    produceAb();      /* Plasma B cell antibody production */
    iB_Ag();        /* B cell + Antigen interaction */
    iAb_Ag();       /* Antibody + Antigen interaction */
    death();        /* Cell death */
    flow();         /* Flow of naive B cells from bone marrow */
    diffusion();      /* Diffusion within the body grid */
```



- Die Hauptsimulationsfunktion enthält eine Schleife, die die Dynamik von jedem Simulationsschritt updated.
- Die Dynamik von jedem der simulierten Prozesse hängt von einer Zahl von Parametern ab, die durch den Benutzer gesetzt werden.

Simulation des Immunsystems XX
IMMSIM Das Programm VI

```
/* Model Parameters (variable names are preceeded with an underscore) */

#define   _xDim       20    /* X dimension of lattices (must be even) */
#define   _yDim       20    /* Y dimension of lattices (must be even) */

#define   _ag       0x0   /* The antigen bitstring */
#define   _divideB    2   /* Number of B cell divisions on activation (3) */
#define   _initB      200   /* Initial number of B cells (1000) */
#define   _pmRatio    0.5   /* Probability that B cell daughter is plasma cell */
#define   _tauAb      3   /* Antibody halflife */
#define   _tauAg      0   /* Antigen halflife */
#define   _tauB       10    /* Naive B cell halflife */
#define   _tauMB      40    /* Memory B cell halflife (50) */
#define   _tauPB      10    /* Plasma B cell halflife */
#define   _numAb      5   /* Number of Ab secreted by plasma B cell per time-step */

#define   _injection    3000  /* How many Ag to inject */
#define   _injection1   0   /* Time of first injection */
#define   _injection2   150   /* Time of second injection */

#define   _bits     8   /* Length if sIg bitstring */
#define   _minMatch     7   /* Minimum number of matching bits required for binding */
#define   _affLevel   0.01  /* Minimum affinity */
#define   _affEnhance   12.5  /* Factor affinity increase for each additional matching bit */

float   IS[_bits+1];      /* Affinity for a p-bit match */
```

- Wegen der Effizienz sind diese Modell-parameter als Konstanten im Code definiert.

Simulation des Immunsystems XXI
IMMSIM Das Programm VII

```
void birth () {

  int x,y;
  struct BCELL *b, *temp;

  /* B cell birth */

  for (x=0; x<_xDim; x++) {
    for (y=0; y<_yDim; y++) {
      b = bLattice[c][x][y];
      while (b != NULL) {

        if (b->divide > 0) {

          b->divide--;

          temp = createB(b);

          if (b->divide == 0) {
            if (COIN(_pmRatio)) {b->type = PLASMA; b->divide = 0;}
            if (COIN(_pmRatio)) {temp->type = PLASMA; temp->divide = 0;}
          }

          addB(temp,x,y);
        }

        b = b->next;
      }
    }
  }
}
```

Abfrage über die noch ausstehenden Zellteilungen (struct BCELL)

Darstellung der neuen B-Zelle siehe dazu die Funktion createB()

Differenzierung der B-Zelle in Plasma- oder Gedächtniszelle

Fortsetzung mit der nächsten B-Zelle

- Funktion **`birth()`** ist bestimmt die Dynamik der Zellteilung.
- Eine B-Zelle, stimuliert durch die Bindung eines Antigens, teilt sich in geringen Umfang. Die Zahl der Teilungen wird durch den Simulationsparameter **`_divideB`** bestimmt.

Simulation des Immunsystems XXII
IMMSIM Das Programm VIII

```
unsigned match (unsigned long A, unsigned long B) {

  unsigned long mVector;
    unsigned long pos = 0x1;
  unsigned match = 0;
  int bit;

  mVector = A^B;

  for (bit=0; bit<_bits; bit++) {
    if (mVector & pos) match++;
    pos <<= 1;
  }

  return match;
}
```

```
struct BCELL *createB(struct BCELL *b) {      /* Creates a single B cell */

  struct BCELL *temp = (struct BCELL *) malloc(sizeof(struct BCELL));

  temp->next = NULL;
  temp->prev = NULL;

  if (b == NULL) {        /* Creating new B cell from bone marrow */

    temp->type = NAIVE;
    temp->divide = 0;
    temp->slg = randInt(0,0x1<<_bits);

  } else {            /* Dividing B cell */

    temp->type = b->type;
    temp->divide = b->divide;
    temp->slg = b->slg;

  }

  /* Pre-calculate the number of matching bits with the antigen */

  temp->match = match(temp->slg,_ag);

  return temp;
}
```

- Während eines Zeitschritts teilt sich eine B-Zelle nur einmal => Definition der Zeitskala der Simulation
- Jede B-Zelle wird mit einem Zähler versehen, um die Zahl der Teilung der Zelle festzuhalten.
- Während der B-Zellteilung gibt es keine anderen Antigeninteraktionen

<= Übergeben wird in birth() der Oberflächenbitstring-Rezeptor und der Antigen-Bit-String für das Matchen.

Simulation des Immunsystems XXIII
IMMSIM Das Programm IX

```
/* Output data */

  printf ("Running time-step %u\n",simTime);

  sprintf(buffer,"%u\t%u\t%u\t%u\t%u\t%u\n",simTime,countB(MEMORY,-1)
  ,countB(MEMORY,8),countB(MEMORY,7),countAb(),countAg());
  fputs(buffer,outFile);
```

Ausgabe der Anzahl der spezifisch matchenden Memoryzellen, der Anti-körper und der Antigene

```
  /* Counting functions */

  unsigned countB (enum bType t, int match) {

    unsigned count = 0;
    int x,y;
    struct BCELL *temp;

    for (x=0; x<_xDim; x++) {
      for (y=0; y<_yDim; y++) {
        temp = bLattice[c][x][y];
        while (temp != NULL) {
          if ((t == ANY || temp->type == t) &&
            (match < 0 || match == temp->match)) count++;
          temp = temp->next;
        }
      }
    }

    return count;
  }

  unsigned countAb () {

    unsigned bt, count = 0;
    int x,y;

    for (bt=0; bt<=_bits; bt++) {
      for (x=0; x<_xDim; x++) {
        for (y=0; y<_yDim; y++) {

          count += abLattice[c][bt][x][y];
        }
      }
    }

    return count;
  }

  unsigned countAg () {

    unsigned count = 0;
    int x,y;

    for (x=0; x<_xDim; x++) {
      for (y=0; y<_yDim; y++) {

        count += agLattice[c][x][y];
      }
    }

    return count;
  }
```

Gezeigt ist der Zugriff auf die Counting functions in der Methode main().

- Nach Ende der Teilungsphase entscheidet die B-Zelle, ob sie eine Plasma- oder eine Gedächtniszelle wird.
- Die Wahl wird durch _pmRatio kontrolliert (_pmRato ist die Angabe der Wahrscheinlichkeit, dass eine B-Zelle eine Plasmazelle wird)

Simulation des Immunsystems XXIV
IMMSIM Das Programm X

```
/*  Some useful probability functions */

#define rand01()    (rand()/(RAND_MAX+1.0))              /* Produces a random number in the interval
[0,1) */

#define randInt(L,H)  ((H>L) ? ((int)(rand01()*((H)-(L)+1)) + (L)) : (L))   /* Produces a random integer in
the interval [L,H] */

#define COIN(p)     (rand01() < p)                  /* Returns true with probability p */

#define PDIE(halfLife)  (1-exp(-LN2/(halfLife)))             /* Returns the probability of death at each
time-step, given halfLife */

#define __max(a,b) (((a) > (b)) ? (a) : (b))
#define __min(a,b) (((a) < (b)) ? (a) : (b))

unsigned long combinatorial(unsigned n, unsigned r) {          /* Returns (n C r) */

  unsigned long i, numerator = 1, denomenator = 1;

  for (i = n; i >= __max(r, n - r) + 1; i--) numerator = numerator * i;

  for (i = __min(r, n - r); i >= 1; i--) denomenator = denomenator * i;

  return (numerator / denomenator);
}
```

- Dieses Programm benutzt die eingebaute rand() Funktion, um Zufallszahlen zu generieren.
- Für eine reale Forschung wird empfohlen, einen qualitativ höherwertigen Generator zu benutzen.

Simulation des Immunsystems XXV
IMMSIM Das Programm XI

- Plasma B-Zell Antikörper-Produktion:
  - Plasma B-Zellen produzieren kontinuierlich Antikörper
  - Für jede existierende Plasma-Zelle fügt die Methode **produceAb() numAb** Antikörper zu den Antikörpern an der gleichen Gitterstelle hinzu, sobald diese von den Plasmazellen produziert wurden

```
void produceAb () {

  int x,y;
  struct BCELL *b;

  /* Plasma B cell production of antibodies */

  for (x=0; x<_xDim; x++) {
    for (y=0; y<_yDim; y++) {
      b = bLattice[c][x][y];
      while (b != NULL) {

        if (b->type == PLASMA) abLattice[c][b->match][x][y] += _numAb;

        b = b->next;
      }
    }
  }
}
```

Produktion der Antikörper mit der spezifischen Rezeptoreigenschaft

Erhöhung der Antikörper-konzentration an der Gitterstelle

Simulation des Immunsystems XXVI
IMMSIM Das Programm XII

- **B-Zell- und Antigeninteraktion:**
  - B-Zellen werden durch die Bindung an Antigene stimuliert.
  - Jede B-Zelle durchläuft eine Interaktion mit einem Antigen während eines jeden Schritts
  - Die Methode iB_Ag() betrachtet alle Paare von B-Zellen und Antigenen(Ag), gelegen an der gleichen Gitterstelle und gibt jedem eine Chance zu interagieren.
  - Wenn sich eine Interaktion ereignet, wird das Antigen aus der Simulation entfernt und die B-Zelle gegen weitere Interaktionen in diesem Zeitschritt geschützt.

Simulation des Immunsystems XXVII
IMMSIM Das Programm XIII

```
void iB_Ag () {   /* B cell + Antigen interaction */

  int x,y;
  unsigned ag;
  struct BCELL *b;

  /* First, randomize the B cell lists.*/
  randomizeB();

  /* Next, do the interaction */

  for (x=0; x<_xDim; x++) {
    for (y=0; y<_yDim; y++) {
      b = bLattice[c][x][y];
      while (b != NULL) {

        for (ag=agLattice[c][x][y]; ag>0; ag--) {

          if (b->type != PLASMA && b->divide == 0 && COIN(IS[b->match])) {

            /* Bound Ag -- stimulate the B cell, remove the antigen */

            b->type = MEMORY;
            b->divide = _divideB;

            agLattice[c][x][y]--;

            break;  /* Move onto the next B cell */
          }
        }

        b = b->next;
      }
    }
  }
}
```

B-Zellen zufalls-verteilt

Interaktions-wahrscheinlichkeit

```
void randomizeB() {

  unsigned i, count, pos = 0;
  int x,y;
  struct BCELL *b;

  for (x=0; x<_xDim; x++) {
    for (y=0; y<_yDim; y++) {

      //Count number of B cells

      b = bLattice[c][x][y]; count = 0;
      while (b != NULL) {count++; b = b->next;}

      if (count > 1) {

        //Set aside temporary storage

        struct BCELL **temp = (struct BCELL **) malloc(count * sizeof(struct BCELL*));

        //Move B cells to array and randomize

        b = bLattice[c][x][y]; count = 0;
        while (b != NULL) {temp[count++] = b; b = b->next;}

        for (i=0; i<count; i++) {
          int rnd = randInt(i,count-1);
          struct BCELL *t = temp[i];
          temp[i] = temp[rnd];
          temp[rnd] = t;
        }

        bLattice[c][x][y] = temp[0];

        for (i=1; i<count-1; i++) {temp[i]->next = temp[i+1]; temp[i]->prev = temp[i-1];}

        temp[0]->next = temp[1]; temp[0]->prev = NULL;
        temp[count-1]->next = NULL; temp[count-1]->prev = temp[count-2];

        free(temp);
      }

    }
  }
}
```

probabilistischer Schritt

Vertauschung

Simulation des Immunsystems XXVIII
IMMSIM Das Programm XIV

- Die Wahrscheinlichkeit mit der ein Interaktionsereignisses eintritt, ist von den Bitstrings der B-Zelle und des Antigens abhängig.
- Definition eines Matchs:
  - Ein Match zwischen 2 Bitsträngen ist die Hamming-Distanz

```
/* Initialize the affinity vector */

 for(m=0; m<=_bits; m++) IS[m] = 0.0;

 IS[_minMatch]=_affLevel;

 for(m=_minMatch+1; m<=_bits; m++) {

   IS[m] = _affEnhance * combinatorial(_bits,m-1) / combinatorial(_bits,m) * IS[m-1];
   if (IS[m] > 1.0) IS[m] = 1.0;
 }
```

Simulation des Immunsystems XXIX
IMMSIM Das Programm XV

Die Interaktionswahrscheinlichkeit zwischen einer B-Zelle und einem Antigen mit einem p-bit-Match, ist durch IS[p] gegeben und entspricht der biologisch gemessenen Affinität.

IS[]:

- Array der Länge _bits+1
- Jede Komponente des Arrays gibt die Interaktions-wahrscheinlichkeit für einen p-bit-Match an, wobei p in der Bandbreite von 0 bis _bits liegt.
  - Die Parameter _minMatch, _affLevel und _affEnhance werden für die Definition des IS[] Arrays in der folgenden Weise benutzt:

Simulation des Immunsystems XXX
IMMSIM Das Programm XVI

- Für p <_minMatch ist IS[p]=0 (Diese Bedingung liefert die Grenze, unter der sich keine Bindungen ereignen können.
- IS[_minMatch] =_affLevel, wobei _minMatch immer > _bits/2 sein sollte, da IS[p] konträr zur biologischen Intuition durch Abnahme startet.
- Die Zunahme der Stärke basierend auf das Vergrössern eines Matchs durch ein Bit ist gleich der Inversen der Typenrate (Binomialkoeffizienten) multipliziert mit dem Parameter _affEnhance:

$$\frac{IS[p+1]}{IS[p]} = \_affEnhance \; X \frac{\binom{\_bits}{p}}{\binom{\_bits}{p+1}} \qquad \text{mit} \quad \binom{a}{b} = \frac{a!}{b!(a-b)!}$$

Simulation des Immunsystems XXXI
IMMSIM Das Programm XVII

- Die Verteilung der B-Zellrezeptorenanzahl mit einer gegebenen Affinität für ein bestimmtes Antigen binomisch verteilt.
- Es gibt perfekte Matches, 1-Bit-Mismatches usw.
- Die Gesamtzahl der B-Zellrezeptortypen, die ein bestimmtes Antigen stimulieren kann, ist gegeben durch:

$$\sum_{b=\_minmatch}^{\_bits} \binom{\_bits}{b}$$

Simulation des Immunsystems XXXII
IMMSIM Das Programm XVIII

```
void iAb_Ag () {    /* Antibody + Antigen interaction */

  int x,y;
  unsigned bt, ab, ag;

  for (x=0; x<_xDim; x++) {
    for (y=0; y<_yDim; y++) {
      for (bt=0; bt<=_bits; bt++) {

        float pBind = IS[bt];

        if (pBind > 0.0) {

          for (ab=abLattice[c][bt][x][y]; ab>0; ab--) {

            for (ag=agLattice[c][x][y]; ag>0; ag--) {

              if (COIN(pBind)) {

                abLattice[c][bt][x][y]--;
                agLattice[c][x][y]--;

                break;  /* Move onto the next antibody */
              }
            }
          }
        }
      }
    }
  }
}
```

Affinität für p-Bit match

gibt true mit Wahrscheinlichkeit pbind zurück

Entfernung des Antikörpers

Entfernung des Antigens

- **Antigen-Antikörper-Interaktion**
  - Ähnlich der B-Zell-Antigeninteraktion
  - Nur Antigen und Antikörper werden aus der Simulation entfernt

Simulation des Immunsystems XXXIII
IMMSIM Das Programm XIX

- ***Zelltod:***
  - Jeder Bestandteiltyp im System hat eine bestimmte Halbwertszeit

```
void death () {

  short die;
  int x,y;
  unsigned bt,ab,ag;
  struct BCELL *b, *temp;

  /* Antibody death */

  for (bt=0; bt<=_bits; bt++)
    for (x=0; x<_xDim; x++)
      for (y=0; y<_yDim; y++)
        for (ab=abLattice[c][bt][x][y]; ab>0; ab--)
          if (_tauAb && COIN(PDIE(_tauAb))) abLattice[c][bt][x][y]--;

  /* Antigen death */

  for (x=0; x<_xDim; x++)
    for (y=0; y<_yDim; y++)
      for (ag=agLattice[c][x][y]; ag>0; ag--)
        if (_tauAg && COIN(PDIE(_tauAg))) agLattice[c][x][y]--;
```

Entfernung des Antikörpers

Entfernung des Antigens

<= Darstellung des Codes für Antigene und Antikörper

Simulation des Immunsystems XXXIV
IMMSIM Das Programm XX

```
/* B cell death */

 for (x=0; x<_xDim; x++) {
  for (y=0; y<_yDim; y++) {
    b = bLattice[c][x][y];
    while (b != NULL) {

      temp = b->next;

      die = 0;

      switch (b->type) {

      case NAIVE:   if (_tauB && COIN(PDIE(_tauB))) die = 1;  break;
      case PLASMA:  if (_tauPB && COIN(PDIE(_tauPB))) die = 1;  break;
      case MEMORY:  if (_tauMB && COIN(PDIE(_tauMB))) die = 1;  break;
      }

      if (die) {  /* Remove the cell from the list */

        if(b->prev) b->prev->next = b->next;
        else bLattice[c][x][y] = b->next;

        if(b->next) b->next->prev = b->prev;

        free(b);
      }

      b = temp;
    }
  }
 }
}
```

Vorbereitung für die Entfernung

Entfernung der B-Zelle

Die spezifische Wahrscheinlichkeit der Halbwertszeit während eines Zeitschritts ist

$$\exp(-\ln(2)/\tau)$$

mit τ = Halbwertszeit des Bestandteils

Simulation des Immunsystems XXXV
IMMSIM Das Programm XXI

```
void flow() {

  /* Add new B cells from bone marrow to balance expected death */

  unsigned bt, num = (unsigned) (0.5 + LN2 * _initB / _tauB);

  for (bt=0; bt<num; bt++) addB(createB(NULL),randInt(0,_xDim-1),randInt(0,_yDim-1));
}
```

Produktion naiver B-Zellen in jedem Zeitschritt

Zufälliges Einfügen in das Gitter

- Um die Anzahl der B-Zellen ohne Infektionseinfluss konstant zu halten, müssen B-Zellen bei Verlust vom Knochenmark nachproduziert werden. Die Anfangszahl an B-Zellen ist _initB und die Halbwertszeit einer naiven B-Zelle ist _tauB.

Simulation des Immunsystems XXXVI
IMMSIM Das Programm XXII

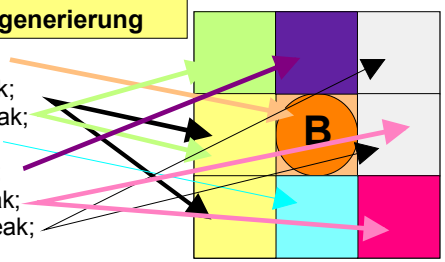
$$\ln(2) \times \frac{\_initB}{\_tauB}$$

Das Knochenmark produziert naive B-Zellen bei jedem Zeitschritt. Diese Zellen werden dem Gitter an zufällig gewählten Positionen durch die Methode flow() eingefügt.

Simulation des Immunsystems XXXVII
IMMSIM Das Programm XXIII

```
void diffusion() {

  int x,y;
  unsigned bt,ab,ag;
  struct BCELL *b, *temp;

  /* Swap the lattices */

  short newLoc, i, old = c;

  c = (c + 1) % 2;

  /* Antibody diffusion -- move from temporary to current lattice */

  for (bt=0; bt<=_bits; bt++)
    for (x=0; x<_xDim; x++) {
      i = (x%2 ? 0 : -1);
      for (y=0; y<_yDim; y++) {
        for (ab=abLattice[old][bt][x][y]; ab>0; ab--) {

          short newLoc = randInt(0,6);
          switch (newLoc) {
            case 0: addAb(bt,x,y);      break;
            case 1: addAb(bt,x-1,i+y);    break;
            case 2: addAb(bt,x-1,i+y+1);  break;
            case 3: addAb(bt,x,y-1);    break;
            case 4: addAb(bt,x,y+1);    break;
            case 5: addAb(bt,x+1,i+y);    break;
            case 6: addAb(bt,x+1,i+y+1);  break;
          }
        }
        abLattice[old][bt][x][y] = 0;
      }
    }
```

Zufallsvariable-generierung



- Diffusion innerhalb des Gitters:
  - In diesem System ist es den Bestandteilen erlaubt zu ihren nächsten Nachbarstellen zu diffundieren. In einem triangulären Gitter gibt es 6 nächste Nachbarnverbindungen.

<= Aus Vereinfachungsgründen ist das Dreiecksgitter rechteckig dargestellt

Simulation des Immunsystems XXXVIII
IMMSIM Das Programm XXIV

```
/* B cell diffusion -- move from temporary to current lattice */

 for (x=0; x<_xDim; x++) {
   short i = (x%2 ? 0 : -1);
   for (y=0; y<_yDim; y++) {
     b = bLattice[old][x][y];
     while (b != NULL) {

       temp = b->next;

       newLoc = randInt(0,6);
       switch (newLoc) {
         case 0: addB(b,x,y);      break;
         case 1: addB(b,x-1,i+y);    break;
         case 2: addB(b,x-1,i+y+1);    break;
         case 3: addB(b,x,y-1);      break;
         case 4: addB(b,x,y+1);      break;
         case 5: addB(b,x+1,i+y);    break;
         case 6: addB(b,x+1,i+y+1);    break;
       }

       b = temp;
     }
     bLattice[old][x][y] = NULL;
   }
 }
}
```

Codestruktur wie bei den Antikörper

```
/* Antigen diffusion -- move from temporary to current lattice */

 for (x=0; x<_xDim; x++) {
   short i = (x%2 ? 0 : -1);
   for (y=0; y<_yDim; y++) {
     for (ag=agLattice[old][x][y]; ag>0; ag--) {

       newLoc = randInt(0,6);
       switch (newLoc) {
         case 0: addAg(x,y);     break;
         case 1: addAg(x-1,i+y);   break;
         case 2: addAg(x-1,i+y+1); break;
         case 3: addAg(x,y-1);   break;
         case 4: addAg(x,y+1);   break;
         case 5: addAg(x+1,i+y);   break;
         case 6: addAg(x+1,i+y+1); break;
       }
     }
     agLattice[old][x][y] = 0;
   }
 }
```

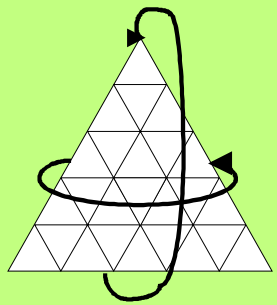
Codestruktur wie bei den Antikörper

- In der Funktion diffusion() ist es jedem Bestandteil erlaubt, zwischen der Bewegung zu einer der nächsten Nachbarstellen oder dem Verbleib auf seiner aktuellen Position zufällig zu wählen.

Simulation des Immunsystems XXXIX
IMMSIM Das Programm XXV

```
/* Functions to add entities to the current lattice incorporating periodic boundary conditions */

void addAb(unsigned bt, int x, int y) {translate(&x,&y); abLattice[c][bt][x][y]++;}

void addAg(int x, int y) {translate(&x,&y); agLattice[c][x][y]++;}

void addB(struct BCELL *b, int x, int y) {

  if (b != NULL) {

    translate(&x,&y);

    b->prev = NULL;

    if (bLattice[c][x][y] == NULL) b->next = NULL;
    else {
      b->next = bLattice[c][x][y];
      bLattice[c][x][y]->prev = b;
    }

    bLattice[c][x][y] = b;
  }
}
```

```
void translate (int *x, int *y) { /* Periodic boundary conditions */

  if (*x<0) *x=_xDim+*x;
  else if (*x>=_xDim) *x=*x%_xDim;
  if (*y<0) *y=_yDim+*y;
  else if (*y>=_yDim) *y=*y%_yDim;
}
```



– Das Problem, das sich durch die Stellen ergibt, die an den Kanten des Gitters gelegen sind, wird durch die Anwendung einer toroidalen Geometrie für das Gitter (d.h. periodische Grenzbedingungen) gelöst.

Simulation des Immunsystems XXXX
IMMSIM Das Programm XXVI

- Diese Simulation zeigt, dass einen 1 zu 1 Übersetzung des biologischen Immunsystems in ein künstliches wegen der 3D-Daten nicht möglich ist.
- Auch die Simulation mittels üblicher Differentialgleichungen (ODE) kombiniert mit numerischer Integration als Alternative ist limitiert:
  - – ODE-Modelle basieren auf genügend grossen Populationen, um die Wahrscheinlichkeit bestimmen zu können. Jedoch jede Zelle hat ihre einzigartige Lebensgeschichte
  - – ODEs geben nur das Durchschnittsverhalten eines Systems wieder.
  - – Modelle eines Immunsystems sind oft nicht linear, was die numerische Integration oder analytische Lösung eines ODE-Modell besonders schwierig macht.
- => gibt es noch weitere Strategien. Das Seminar wird es zeigen

Vielen Dank für die Aufmerksamkeit