ANHANG A: MATLAB-CODE

Die MATLAB-Files sind in drei Unterverzeichnisse verteilt. Jedes dieser enthält ein Hauptfile und benötigt dazu seine eigenen Funktionen. Nachfolgend wir jedes Hauptfile kurz beschrieben und es wird erläutert, welche Funktionen dazu benötigt werden:

- *entzerrer.m*: Ist das Hauptfile für die Simulation des Entzerrers (Kap. 5). Sämtliche sinnvollen Änderungen werden in diesem File vorgenommen. Da die Simulation sehr universell ist, kann sie auch für andere Systeme mißbraucht werden. Roll-Off-Faktor, Abtastfrequenz, Wertigkeit, Nyquistbandbreite und sogar den Signal-Rauschabstand kann bei der Initialisierung verändert werden. Diese File benötigt folgende Funktionen:
  - *augendiagramm.m*
  - *raisedcos.m*
  - *rls.m*
  - *rootcos.m*
  - *stoss.m*
  - *verzoeg.m*

- *simulation.m*: Mit diesem File kann alles simuliert werden, was in Kap. 6 dargestellt ist (außer der Verzögerung der Stossantworten). Dieses File ruft auch die verschiedenen SIMULINK-Modelle auf. Die Simulationszeiten können verändert werden. Die Resultate der BER-Simulation ist im File *resultate.m* vorhanden. Das File *simulation.m* benötigt folgende Funktionen und Modelle:
  - *augendiagramm.m*
  - *ber.m*
  - *detektor.m*
  - *nutzdaten.m*
  - *pll_1.m*
  - *pll_2.m*
  - *quantisier_4.m*
  - *raisedcos.m*
  - *rls.m*
  - *rootcos.m*
  - *verzoeg.m*
  - *entzerrt_2.mdl*
  - *entzerrt_4.mdl*
  - *entzerrt_sch.mdl*
  - *verzerrt_2.mdl*

- *verz_sim.m*: In diesem File wird die Verzögerung der Stossantworten (und damit des Eingangssignals), wie in Kap. 6 beschrieben, simuliert. Dieses File benötigt folgende Funktionen:
  - *augendiagramm.m*
  - *int.m*
  - *rootcos.m*
  - *stoss.m*

Nun werden zuerst die einzelnen MATLAB-Funktionen und dann die SIMULINK-Modelle beschrieben (in alphabetischer Reihenfolge):

- *augendiagramm.m*: Mit dieser Funktion kann ein Augendiagramm eines beliebigen Signals gezeichnet werden. Der Signalvektor, die Verschiebung, der Startwert und die Abtastzeit müssen übergeben werden. Der Startwert ist nützlich, um allfällige Einschwingvorgänge zu unterdrücken. Diese Funktion hat keinen Rückgabewert.
- *ber.m*: Mit Hilfe dieser Funktion kann man die BER von zwei Datenströmen ausrechnen. Diese Funktion liefert die Anzahl Fehler, die Position derer und berechnet die BER.
- *detektor.m*: Simuliert den Phasendetektor (eigener Algorithmus), wie er in Kap. 6 beschrieben wurde. Diese Funktion wird als „MATLAB-Funktion“ von SIMULINK her aufgerufen.
- *int.m*: Interpoliert eine Stossantwort zwischen den Abtastwerten mit der Spline-Funktion. Diese Funktion liefert einen Vektor, der int mal mehr Werte aufweist, sowie eine Matrix, die verschobenen Stossantworten beinhaltet.
- *nutzdaten.m*: Entfernt die Nullen aus einem Datenstrom.
- *pll_1.m*: Simuliert den Null-Wert-PLL nach Sailer. Diese Funktion wird als „MATLAB-Function“ von SIMULINK her aufgerufen.
- *pll_2.m*: Simuliert den Schwellwert-PLL (eigener Algorithmus). Diese Funktion wird als „MATLAB-Function“ von SIMULINK her aufgerufen.
- *quantisier_4.m*: Quantisiert das Eingangssignal auf vier Levels. Die Schwellen können eingestellt werden.
- *raisedcos.m*: Diese Funktion erlaubt die Berechnung eines Raised-Cosine-Filters. Abtastfrequenz, Nyquistbandbreite, Roll-Off-Faktor, Filterordnung und die Anzahl Frequenzstützpunkte müssen dabei übergeben werden. Zurückgegeben werden die berechneten Filterkoeffizienten nach remez und nach firls. Erfahrungsgemäß liefert firls die bessere Approximation dieses Problems. Beide Resultate werden graphisch dargestellt.
- *rls.m*: Diese Funktion adaptiert (trainiert) ein FIR-Filter. Die Anzahl Filterkoeffizienten, den Filtereingang, die gewünschte Antwort und die Verschiebung muß übergeben werden. Das p0 ist die Initialisierung der Matrix und wird auf etwa 1000 gesetzt (dies ergab bei den Tests gute Ergebnisse). Diese Funktion hat vier Rückgabewerte: Das Fehlersignal, die Koeffizienten im Verlaufe der Adaption, die endgültigen Filterkoeffizienten und das Ausgangssignal des Filters.
- *rootcos.m*: Diese Funktion ist mit dem Handling identisch mit *raisedcos.m*, nur wird ein Root-Raised-Cosine-Filter berechnet.
- *stoss.m*: Erzeugt die Eingangssignale für das FIR-Filter in Form von Konecker-Delta Stößen. Wertigkeit, Anzahl Stöße und die Verschiebung der Stossantworten müssen dazu übergeben werden. Die Funktion liefert dann einen zufällig erzeugten Signalvektor.

- *verzoeg.m*: Verzögert ein Datensignal um ganze Werte, die ersten Werte werden mit Null ergänzt
- *entzerrt_2.mdl*: Enthält das SIMULINK-Modell der entzerrten, zweiwertigen Übertragung mit dem Null-Wert-PLL (Teil 2 von *simulation.m*).
- *entzerrt_4.mdl*: Enthält das SIMULINK-Modell der entzerrten, vierwertigen Übertragung mit dem Phasendetektor (Teil 4 von *simulation.m*).
- *entzerrt_sch*: Enthält das SIMULINK-Modell der entzerrten, zweiwertigen Übertragung mit dem Schwellwert-PLL (Teil 3 von *simulation.m*).
- *verzerrt_2.mdl*: Enthält das SIMULINK-Modell der verzerrten, zweiwertigen Übertragung mit dem Null-Wert-PLL (Teil 1 von *simulation.m*).