/\* Takes study data ({@link mrmc.core.InputFile#keyedData}, {@link mrmc.core.InputFile#truthVals}) \* and creates data for {@link mrmc.core.CovMRMC} \* --t-matrices: reader scores \* ---- {@link #t0 modAA}, {@link #t0 modAB}, {@link #t0 modBB}: signal-absent scores [Nnormal][Nreader][2 modalities] \* ---- {@link #t1\_modAA}, {@link #t1\_modAB}, {@link #t1\_modBB}: signal-present scores [Ndisease][Nreader][2 modalities] \* --d-matrices: study design \* ---- {@link #d0\_modAA}, {@link #d0\_modAB}, {@link #d0\_modBB}: signal absent indicator [Nnormal ][Nreader][2 modalities] {@link #d1\_modAA}, {@link #d1\_modAB}, {@link #d1\_modBB}: signal present scores [Ndisease][Nreader][2 modalities] \* --fully crossed status \* ---- {@link #fullyCrossedA}, {@link #fullyCrossedB}, {@link #fullyCrossedAB} \* CALLED BY: {@link mrmc.gui.InputFileCard.varAnalysisListener} \*/ t0\_modAB = new double[(int) Nnormal][(int) Nreader][2]; t1\_modAB = new double[(int) Ndisease][(int) Nreader][2]; t0\_modBB = new double[(int) Nnormal][(int) Nreader][2]; t1\_modAA = new double[(int) Ndisease][(int) Nreader][2]; t1\_modAA = new double[(int) Ndisease][(int) Nreader][2]; t0\_modAA = new int[(int) Nnormal][(int) Nreader][2]; d1\_modAA = new int[(int) Ndisease][(int) Nreader][2]; d1\_modBB = new int[(int) Ndisease][(int) Nreader][2]; d0\_modAB = new int[(int) Nnormal][(int) Nreader][2]; d1\_modAB = new int[(int) Ndisease][(int) Nreader][2]; int PresentModA; int PresentModB; // signal-absent case counter // signal-present case counter // reader counter InputFileStat.keyedD ata.keySet()) End End for // For all readers and cases, determine which had observations End for InputFileStat.keyedData. InputFileStat.keyedD ata.get(r).keySet()) get(r).get(c).containsKey .get(c).containsKey(c)? (modA) Yes No ScoreModA = -1000000; ScoreModB = -1000000; PresentModA = 0; ScoreModB = -1000000; PresentModB = 0; fullyCrossedB = false; InputFileStat.keyedData.get (r).get(c).containsKey(modB) coreModB = InputFileStat.keyedData.get(r).get(c).get(modB); InputFileStat.truthVals.get(c) == 0 No-// Fill in the score and design matrices Yes t0\_modAB[m][k][1] = ScoreModA; t0\_modAA[m][k][0] = ScoreModA; t0\_modAA[m][k][1] = ScoreModA; t0\_modBB[m][k][0] = ScoreModB; t0\_modBB[m][k][1] = ScoreModB; t1\_modAB[n][k][1] = ScoreModA; t1\_modAA[n][k][0] = ScoreModA; t1\_modAA[n][k][1] = ScoreModA; t1\_modBB[n][k][0] = ScoreModB; t1\_modBB[n][k][1] = ScoreModB; d0\_modAB[m][k][0] = PresentModA; d0\_modAB[m][k][1] = PresentModB; d0\_modAA[m][k][0] = PresentModA; d0\_modAA[m][k][1] = PresentModA; d0\_modBB[m][k][0] = PresentModB; d1\_modAB[n][k][0] = PresentModA; d1\_modAB[n][k][1] = PresentModB; d1\_modAA[n][k][0] = PresentModA; d1\_modAA[n][k][1] = PresentModB; d1\_modBB[n][k][0] = PresentModB; d0\_modBB[m][k][1] = PresentModB; d1\_modBB[n][k][1] = PresentModB;