# 4 Turbo base stations

Emre Aktas, Defne Aktas, Stephen Hanly, and Jamie Evans

## 4.1 Introduction

Cellular communication systems provide wireless coverage to mobile users across potentially large geographical areas, where base stations (BSs) provide service to users as interfaces to the public telephone network. Cellular communication is based on the principle of dividing a large geographical area into *cells* which are serviced by separate BSs. Rather than covering a large area by using a single, high-powered BS, cellular systems employ many lower-powered BSs each of which covers a small area. This allows for the reuse of the frequency bands in cells which are not too close to each other, increasing mobile user capacity with a limited spectrum allocation.

Traditional narrowband cellular systems require the cochannel interference level to be low. Careful design of frequency reuse among cells is then crucial to maintain cochannel interference at the required low level. The price of low interference, however, is a low frequency reuse factor: only a small portion of the system frequency band can be used in each cell. More recent wideband approaches allow full frequency reuse in each cell, but the cost of that is increased intercell interference. In both approaches, the capacity of a cell in a cellular network, with six surrounding cells, is much less than that of a single cell operating in an intercell interference-free environment. In this chapter, we survey an approach that allows the cell with neighbors to achieve essentially the same capacity as the interference-free cell.

In a conventional cellular system, each mobile user is serviced by a single BS, except for the soft-handoff case – a temporary mode of operation where the mobile is moving between cells and is serviced by two base stations. A contrasting idea is to require each mobile station to be serviced by all BSs that are within its reception range. In this approach all the BSs in the cellular network are components of a single transceiver with distributed antennas, an approach known as "network multiple-input multiple-output (MIMO)."

*Cooperative Cellular Wireless Networks*, eds. Ekram Hossain, Dong In Kim, and Vijay K. Bhargava. Published by Cambridge University Press. 

Network MIMO requires cooperation between BSs. On the uplink, the BSs must cooperate to jointly decode the users, whilst on the downlink, the BSs must cooperate to jointly broadcast signals to all the users in the network. This approach may appear unrealistically complex, but information-theoretic studies have highlighted the potentially huge capacity gains from such an approach [23, 49, 59]. In a nutshell, these works (and others) have shown that such cooperation effectively eliminates intercell interference. In other words, the per-cell capacity of a network of interfering cells is roughly the same as a non-interfering system where the cells are isolated and do not interfere at all (in fact, there is a diversity advantage for the interfering system, which means its capacity is higher than the capacity of the isolated cell model). In the network of interfering cells there is no wasted interference: all received signals contain useful information. Crucially, to obtain this advantage, it is *necessary* for there to be intercell interference: it was shown in [23, 59] that full frequency reuse in each cell is *required* in order to achieve the full information-theoretic capacity. This is in contrast to the conventional cellular model with single cell processing which usually requires fractional frequency reuse.

The question then arises: how can such cooperation be realized in practice? It is natural to conceive first a centralized system in which a central processor is connected to all the base stations, so that the network is operated as a single cell MIMO system, but with distributed antennas. Such an architecture is, however, expensive to build, has a single point of failure, and does not satisfactorily address issues of complexity and delay. A more feasible and desirable solution is to distribute the processing among the base stations. In this chapter we present distributed BS cooperation methods for joint reception and transmission, which allow the desired network MIMO behavior to emerge in a *distributed* manner.

For distributed processing, communication among the BSs is mandatory. The desired properties of a feasible distributed method are: (1) communication should only be required between neighboring BSs, as opposed to message passing among all BSs, and (2) the processing per BS and message passing delay should remain constant with increasing network size. In this chapter, we survey an approach to BS cooperation (and provide new results for this approach) based on a graphical model of the network-MIMO communication processes. In essence, we show that both uplink and downlink modes of communication reduce to belief propagation on graphs derived from the way BSs are interconnected in the backhaul, and from the signal propagation between BSs and mobiles, and vice versa, across the air interface.

To give a simple picture of what we mean by message passing between BSs, consider a cellular network where the BSs and the cells are placed on a line. In this model, every cell has two neighboring cells. Although this simple model is far from being realistic, it provides a framework where the main concepts of distributed processing with message passing can be developed and explained, and it can then be generalized to less restrictive models. The one-dimensional cellular array is illustrated in Figure 4.1.

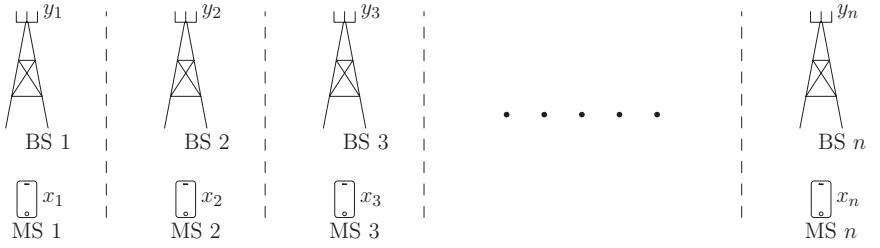



Figure 4.1. Linear cellular array. The cells are positioned on a line. Each cell has one active mobile station (MS). Dashed lines show boundaries between cells. At cell $i$, $x_i$ and $y_i$ represent the transmitted symbol and the received signal.

Let $x_i$ denote the data symbol transmitted by mobile station (MS) $i$ and $y_i$ denote the channel output observed at BS $i$. In the linear cellular array model, the relationship between the transmitted symbols and the received signals is

$$y_i = h_i(-1)x_{i-1} + h_i(0)x_i + h_i(1)x_{i+1} + z_i, \tag{4.1}$$

where $h_i(j)$ is the channel coefficient from MS $i + j$ to BS $i$, and $z_i$ is the additive Gaussian noise with variance $\sigma^2$. We assume that the channel coefficients $h_i(j)$ and the noise variance are known at BS $i$. For convenience, for the cells at the edges of the network, add dummy symbols $x_0$ and $x_{n+1}$, and set the corresponding $h_i(j)$s to zero. The signal model for the one-dimensional cellular array model is depicted in Figure 4.2.

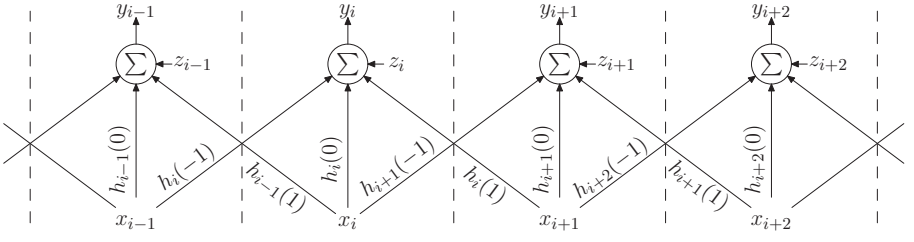



Figure 4.2. Linear cellular array signal model. The symbol transmitted in one cell is received at that cell, and also in the two neighboring cells (one neighboring cell if it is one of the two edge cells).

In the traditional single-cell processing (SCP) approach, BS $i$ tries to detect symbol $x_i$ based on $y_i$ alone. Using a frequency-reuse factor of 1/2 avoids the intercell interference, but this halves the capacity of the system. With full frequency reuse, MS $i$ receives interference from MSs $i - 1$ and $i + 1$, as is clear in (4.1). One could treat this interference as Gaussian noise, and use a mis-matched decoder to decode the desired signal, but information theory tells us that intercell interference can be completely eliminated via multicell processing (MCP) [23, 49, 59].

MCP requires cooperation between the BSs, but how much cooperation is required in the simple model we are considering here? At first sight, it might seem sufficient for BS $i$ to use $(y_{i-1}, y_i, y_{i+1})$ in the detection of symbol $x_i$, as these are the only outputs to which $x_i$ actively contributes. This is not the case, but it is certainly true that BS $i$ can do a much better job of detecting $x_i$ in this scenario. The BS's task is first to compute the conditional distribution $p(x_i|y_{i-1}, y_i, y_{i+1})$ and then to pick the maximum a posteriori estimate for $x_i$. One approach to realize this detection strategy would be for each BS to pass the observed channel output to its immediate neighbors: thus, BS $i$ sends $y_i$ to BSs $i-1$ and $i+1$ respectively. This strategy involves one single message passing between adjacent BSs.

Considering this further, however, we see that intercell interference has not been eliminated after a single message passing step. For example, $y_{i-1}$ receives a contribution from data symbol $x_{i-2}$ and the uncertainty in $x_{i-2}$ must be accounted for in the above probabilistic model. Again, it could be treated as Gaussian noise, or it could be modeled more accurately than that, depending on what is measured or known by the BSs, and what information is passed from one to the other. For example, BS $i$ *may* know the constellations from which the interfering symbols $x_{i-2}$ and $x_{i+2}$ have been chosen. The BS may also have phase information (the coherent case) or the phase may be unknown (incoherent). The exact model used by BS $i$ depends on which particular assumptions best describe the real-world scenario, but in all these possible models, intercell interference remains after one message passing step in the effect of the unknown symbols $x_{i-2}$ and $x_{i+2}$, which cannot be reliably detected.

The above interference model may remind the reader of standard intersymbol interference (ISI) channels that arise in frequency-selective digital communication scenarios. Such models are linear, and if we assume in addition that the a priori distributions on the input symbols are Gaussian, then the optimal equalizer is to apply the matched filter (in this case, the linear minimum mean squared error (LMMSE) filter) to the observed symbols $y_1, y_2, ..., y_n$. This makes it clear that it is not optimal for BS $i$ to have access only to $(y_{i-1}, y_i, y_{i+1})$: to be optimal, BS $i$ requires all the channel outputs $y_1, y_2, ..., y_n$, as well as all the channel gains, and information about the a priori distributions on the symbols. With that information, it can apply the optimal filter, and obtain an optimal estimate of $x_i$. In other words, there is a system-wide coupling of the interference between cells. This approach might be called centralized MCP .

The problem with centralized MCP is that it requires a huge amount of message passing. All BSs require global channel knowledge in order to each apply the globally optimal filter. Note, however, that distributed methods can be used in ISI equalization. In the Gaussian case, the LMMSE estimates can be obtained by the recursive Kalman smoother . In the case of discrete input constellations, the maximum a posteriori (MAP) detector can be obtained by the forward–backward or BCJR algorithm [8] . Such methods are special cases of Bayesian estimation for graphical models. This suggests the idea of representing the

cellular network by a graphical model, and obtaining distributed versions of MCP that do not require each BS to obtain the complete global channel state information (CSI). Further, these methods will allow us to investigate how well performance improves with the number of message passing steps. For example, in some scenarios, we will see that a single message passing step is sufficient to get most of the gains of MCP, whereas in other scenarios, many more message passing steps are required.

The challenge in the area of turbo BSs is to distribute the computations of the conditional distributions of the $x_i$s, so that they can be obtained by message passing between neighboring BSs only. We do this for the uplink in Sections 4.3, and 4.4. In Section 4.5, we apply similar ideas to the downlink broadcast channel problem in which the BSs are sending data symbols to the MSs. To initiate this study, our first step will be to review message passing and belief propagation methods in a more generic framework, and then to apply the results from this theory to the cellular models of interest in this chapter.

## 4.2 Review of message passing and belief propagation

The distributed algorithms presented in this chapter are built on the key concepts of factor graphs and the sum-product algorithm. We begin with a brief review of these concepts.

The use of iterative, or turbo, receiver methods defined on graphs has become an important focus of research in communications since the success of turbo codes and the rediscovery of low-density parity-check codes. Both the turbo decoder [38] and the low-density parity-check code decoder [20] are instances of belief propagation on associated graphs.

A factor graph is a graphical representation on which message passing algorithms are defined. There are at least two other popular graphical representations employed in the communications literature. Firstly, there are graphs on which codes are defined. These graphs represent sets of constraints which describe a code and include Tanner graphs [51], Tanner–Wiberg–Loeliger (TWL) graphs [58], and Forney graphs [17]. These graphs also provide iterative decoding of the associated codes via message passing algorithms. Secondly, there are probabilistic structure graphs including Markov random fields [29] and Bayesian networks [44]. These graphs represent statistical dependencies among a set of random variables. Markov random fields are based on local Markov properties, whereas Bayesian networks are based on causal relationships among the variables and factoring the joint distribution into conditional and marginal probability distribution functions. The message passing algorithms defined on these structures provide methods of probabilistic inference: compute, estimate, and make decisions based on conditional probabilities given an observed subset of random variables.

Factor graphs are not specifically based on describing code constraints or probabilistic structures. They indicate how a joint function of many variables factors into a product of functions of smaller sets of variables. They can be used, however, for describing codes and decoding codes, and in describing probabilistic models and statistical inference. In fact, factor graphs are more general than Tanner, TWL, and Forney graphs for describing codes [34], and they are more general than Markov random fields and Bayesian networks in terms of expressing factorization of a global distribution [19].

### 4.2.1 Factor graph review

In this subsection, we provide just enough review for the uninitiated reader to be able to grasp the BS cooperation material presented in this chapter. For further information, the reader may refer to [30] and the excellent tutorials [32, 33]. The reader experienced in factor graphs may skip this section.

Let $g(x_1, x_2, \ldots, x_n)$ be a function of variables $x_1, \ldots, x_n$, where for each $i$, $x_i$ takes on values in a set $A_i$.

*Definition of marginal function and summary notation*
We are interested in a numerically efficient computation of the marginal function

$$g_i(x_i) = \sum_{\sim\{x_i\}} g(x_1, x_2, \ldots, x_n) \tag{4.2}$$

for some $i$. The right hand side of (4.2) denotes the summation for $x_i$ of function $g$ as defined in [30]: for each $a \in A_i$ the value of $g_i(a)$ is obtained by summing the value of $g(x_1, x_2, \ldots, x_n)$ over all $(x_1, \ldots, x_n) \in A_1 \times \cdots \times A_n$ such that $x_i = a$. For example, for $n = 3$, the summation for $x_2$ of $g$ is

$$g_2(x_2) = \sum_{\sim\{x_2\}} g(x_1, x_2, x_3) = \sum_{x_1 \in A_1} \sum_{x_3 \in A_3} g(x_1, x_2, x_3).$$

*Relationship to the APP*
For probabilistic models, the computation of the marginal in (4.2) is related to the computation of the a posteriori probability (APP), a quantity of particular interest to us in this chapter. Let $(x_1, \ldots, x_n)$ denote the realization of some random variables in a probabilistic model, let $(y_1, \ldots, y_m)$ denote some observed variables in the model, and let $p(x_1, \ldots, x_n, y_1, \ldots, y_m)$ denote the joint distribution. Taking a given $(y_1, \ldots, y_m)$ as fixed, (i.e., observed) define the global function $g$:

$$g(x_1, \ldots, x_n) = p(x_1, \ldots, x_n, y_1, \ldots, y_m). \tag{4.3}$$

Typically, $g$ is factorized into two as

$$g(x_1, \ldots, x_n) = p(y_1, \ldots, y_m | x_1, \ldots, x_n) p(x_1, \ldots, x_n),$$

where the first term is the likelihood function and the second term is the a priori distribution of $(x_1, \ldots, x_n)$. Depending on the probabilistic model, these two factors themselves are further factorized. The APP of $x_i$ for any desired $i \in \{1, \ldots, n\}$ is proportional to the marginal of $g$ for $x_i$:

$$p(x_i|y_1, \ldots, y_m) \propto g_i(x_i), \tag{4.4}$$

where $g_i(x_i)$ is the marginal of the joint distribution in (4.3), and the notation "$\propto$" means "proportional to", i.e., the right hand side of "$\propto$" is scaled by a constant to obtain the left side. If the left hand side is a probability function, this scaling constant can be found using the fact that this function adds up to unity over all possible values of its argument.

*Definition of factor graph*

Suppose that $g(x_1, \ldots, x_n)$ is in the form of a product of *local functions* $f_j$:

$$g(x_1, \ldots, x_n) = \prod_{j=1}^{J} f_j(X_j), \tag{4.5}$$

where $X_j$ is a subset of $\{x_1, \ldots, x_n\}$, and the function $f_j(X_j)$ has the elements of $X_j$ as arguments.

A factor graph represents the factorization of $g(x_1, \ldots, x_n)$ as in (4.5). The corresponding factor graph has two types of nodes: variable nodes and factor nodes. For each variable $x_i$ there is a variable node shown by a circled $x_i$, and for each local function $f_j$ there is a factor node shown by a solid square in the graph. Thus there are $n$ variable nodes and $J$ factor nodes in the graph. There is an undirected edge connecting variable node $x_i$ to factor node $f_j$ if and only if $x_i$ is an argument of $f_j$. Thus connections are only between variable and factor nodes; two factor nodes are never connected, and two variable nodes are never connected. We define the neighbors of a variable node to be those factor nodes to which it is directly connected in the graph. We correspondingly define the neighbors of a factor node to be those variable nodes in the graph to which it is directly connected.

*Definition of sum–product algorithm*

The goal of the sum–product algorithm is to obtain the marginal function in (4.2) for some $i \in \{1, \ldots, n\}$. This is done in a numerically efficient manner, based on the factorization in (4.5) using the distributive law to simplify the summation. The algorithm is defined in terms of messages between connected factor and variable nodes. A message from node $a$ to node $b$ is computed based on previously received messages at node $a$ from all its neighbors except for node $b$. A message from variable node $x_i$ to factor node $f_j$ is a function with argument $x_i$ that can take on values in $A_i$. A message from factor node $f_j$ to variable node $x_i$ is also a function of $x_i$. After the messages from all nodes propagate through the factor graph in a sequential manner, at termination, the incoming messages

at desired variable nodes are combined in order to obtain the associated marginal function. The rules for message updates are given below.

Message from variable node $x$ to factor node $f$:

$$\mu_{x-f}(x) = \prod_{h \in n(x) \backslash \{f\}} \mu_{h-x}(x). \tag{4.6}$$

Message from factor node $f$ to variable node $x$:

$$\mu_{f-x}(x) = \sum_{\sim\{x\}} \left( f(n(f)) \prod_{y \in n(f) \backslash \{x\}} \mu_{y-f}(y) \right), \tag{4.7}$$

where

$n(x)$ : set of all factor node neighbors of variable node $x$ in the factor graph,

$n(x)\backslash\{f\}$ : set of all neighbors of $x$ except for $f$,

$n(f)$ : set of all variable node neighbors of factor node $f$ in the factor graph.

We make the following observations on the messages in the sum–product algorithm. The computations done by variable nodes in (4.6) are a simple multiplication of incoming messages, whereas the computations done by the factor nodes in (4.7) are more complex. A variable node of degree 2 (i.e., a node with two neighbors) simply replicates the message received on one edge onto the other edge. A factor node of degree 1 simply outputs the function of the variable that it is connected to as the message.

The computation typically starts at the leaf nodes of the factor graph. Each leaf variable node sends a trivial identity function. If the leaf node is a factor node, it sends a description of $f$. If the computation is started from nonleaf nodes, it is assumed that it has received trivial identity messages during initiation. Each node remains idle until it receives all required messages based on which it can compute outgoing messages.

To terminate the computations, the messages are combined at the desired variable nodes. The rule for combining messages at a variable node is to take the product of all incoming messages:

$$\mu_x(x) = \prod_{h \in n(x)} \mu_{h-x}(x). \tag{4.8}$$

Equivalently, $\mu_x(x)$ can be computed as the product of the two messages that were passed in opposite directions over any single edge incident on $x$:

$$\mu_x(x) = \mu_{f-x}(x)\mu_{x-f}(x) \qquad \text{for any } f \in n(x). \tag{4.9}$$

If the factor graph is a tree, then $\mu_x(x)$ will be the marginal function $g(x)$ defined in (4.2). If the factor graph has loops, then the message passings can be repeated, and at termination $\mu_x(x)$ will be an approximation of the marginal

function $g(x)$. In many cases, scaled versions of the messages are computed, which results in a $\mu_x(x)$ scaled by a constant. Thus the final $\mu_x(x)$ is obtained after a proper normalization.

*Definition of $[P]$ notation*
If $P$ is a Boolean proposition involving some set of variables, then $[P]$ is the $\{0,1\}$-valued truth function

$$[P] = \begin{cases} 1, & \text{if } P \text{ is true,} \\ 0, & \text{if } P \text{ is false.} \end{cases} \tag{4.10}$$

### 4.2.2 Factor graph examples

*Example 1 Hidden Markov model*
Consider a probabilistic model where we have the states vector $s = (s_1, s_2, \ldots, s_n)$ and output variables vector $u = (u_1, u_2, \ldots, u_n)$. The states $s_1, \ldots, s_n$ form a Markov chain, and the transition from $s_{i-1}$ to $s_i$ produces an output variable $u_i$.

The local function $T_i$ computes the conditional probability of transitioning from $s_{i-1}$ to $s_i$, and the output $u_i$:

$$T_i(s_{i-1}, u_i, s_i) = p(s_i|s_{i-1})p(u_i|s_i, s_{i-1}) \qquad \text{for } i = 1, \ldots, n. \tag{4.11}$$

In several examples, $u_i$ is a function of only $s_i$, so in those examples,

$$T_i(s_{i-1}, u_i, s_i) = p(s_i|s_{i-1})[u_i = d(s_i)],$$

where $d$ is the function that determines $u_i$.

Corresponding to each output variable $u_i$ is the "noisy" observation $y_i$, where the relationship between the output variable and its observation is characterized by the conditional distribution $p(y_i|u_i)$. The global function of $(s, u)$ is

$$\begin{aligned} g(s,u) &= p(y|s,u)p(s,u) \\ &= \left(\prod_{i=1}^{n} p(y_i|u_i)\right)\left(\prod_{i=1}^{n} T_i(s_{i-1}, u_i, s_i)\right). \end{aligned} \tag{4.12}$$

Note that $y$ is fixed for any realization of observation, so we consider $g(s,u)$ to be a function of $(s,u)$ only, and regard $y$ as a vector of parameters.

The factor graph corresponding to the factorization in (4.12) is given in Figure 4.3 for $n = 3$. The dummy nodes added in this graph do not alter the function $g$ nor the resulting algorithm, but they allow a convenient description of the algorithm. For $T_1$, the state transition from $s_0$ to $s_1$ is independent of $s_0$.

During initialization, each pendant factor node sends the messages, which are their function descriptions to their corresponding variable nodes. Then, since the corresponding variable nodes are all of order 2, they replicate the messages at the other edge. Afterwards, forward ($s_{i-1}$ to $s_i$ for $i = 1, \ldots, n$) and backward ($s_i$ to

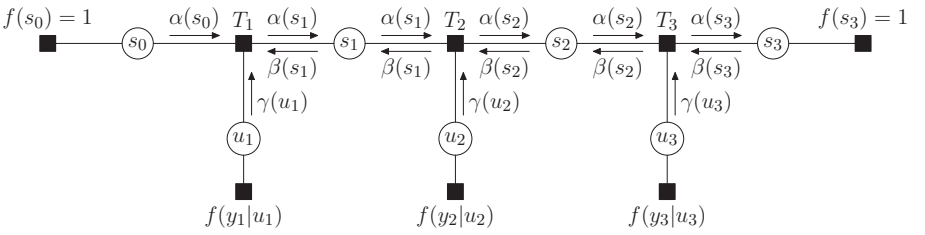



Figure 4.3. Factor graph for hidden Markov model for $n = 3$. Dummy nodes $f(s_0)$, $s_0$, and $f(s_3)$ are added to handle the initialization of the algorithm at the edges of the Markov chain. Since the variable nodes in this graph have degree 2, they simply replicate the message received on one edge on the other edge.

$s_{i-1}$ for $i = n, \ldots, i+1$) message passing occurs along the chain. The resulting algorithm is known as the forward–backward or BCJR algorithm [8]. In the literature, the message $\mu_{u_i - T_i}(u_i)$ is denoted by $\gamma(u_i)$, the message $\mu_{T_i - s_i}(s_i)$ is denoted by $\alpha(s_i)$, and the message $\mu_{T_i - s_{i-1}}(s_{i-1})$ is denoted by $\beta(s_{i-1})$. Using that notation, at initialization, we have

$$\begin{aligned} &\gamma(u_i) = p(y_i|u_i) = f(y_i|u_i) \qquad \text{for } i = 1, \ldots, n, \\ &\alpha(s_0) = 1, \\ &\beta(s_n) = 1. \end{aligned}$$

Then the forward recursion is computed as the message from $T_i$ to $s_i$, using (4.7):

$$\alpha(s_i) = \sum_{\sim\{s_i\}} T_i(s_{i-1}, u_i, s_i)\alpha(s_{i-1})\gamma(u_i) \qquad \text{for } i = 1, \ldots, n \tag{4.13}$$

and the backward recursion is computed as the message from $T_i$ to $s_{i-1}$

$$\beta(s_{i-1}) = \sum_{\sim\{s_{i-1}\}} T_i(s_{i-1}, u_i, s_i)\beta(s_i)\gamma(u_i) \qquad \text{for } i = n, \ldots, 2. \tag{4.14}$$

This is the general form of the forward–backward algorithm. For different specific cases, the local functions are different but the general structure of the algorithm is the same, outlined by the forward and backward recursions in (4.13) and (4.14).

After the forward and backward recursions are complete, at termination, for each state variable node $s_i$, the incoming messages are combined as

$$\mu_{s_i}(s_i) = \alpha(s_i)\beta(s_i) \qquad \text{for } i = 1, \ldots, n. \tag{4.15}$$

Since the factor graph is a tree, $\mu_{s_i}(s_i)$ is, in fact, the true marginal $g_i(s_i)$ and a scaled version of the APP $p(s_i|y)$. This model is directly applicable to the uplink of the simple one-dimensional cellular network that we examine in Section 4.3. It is the simplest model of turbo BS cooperation that we encounter in this chapter.

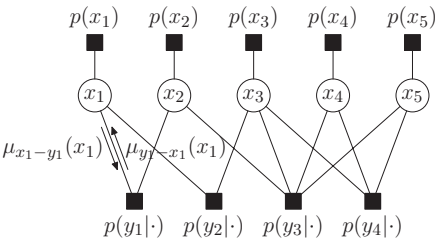



Figure 4.4. Factor graph for the interference channel model for $n = 5$, $m = 4$, $n_{y_1} = \{x_1, x_2\}$, $n_{y_2} = \{x_1, x_3\}$, $n_{y_3} = \{x_2, x_3, x_4, x_5\}$, and $n_{y_4} = \{x_3, x_4, x_5\}$. The notation $p(y_i|\cdot)$ refers to the conditional distribution of $y_i$ given the neighbor variable nodes: $p(y_i|\cdot) = p(y_i|n_{y_i})$. In the following sections, the prior distribution factor nodes, $p(x_i)$, will not be shown in the graphs.

*Example 2 Interference channel*

Consider a channel with $n$ input variables $x = \{x_1, \ldots, x_n\}$ and $m$ output variables $y = \{y_1, \ldots, y_m\}$. Each output variable is a noisy observation of a linear combination of the elements in a subset of the inputs, indexed by $n_i \subset \{1, \ldots, n\}$,

$$y_i = \sum_{j \in n_i} h_{i,j} x_j + z_i, \tag{4.16}$$

where $h_{i,j}$ is the complex channel coefficient of input $x_j$ at the channel output $y_i$, and $z_i$ is the additive white circularly symmetric complex Gaussian noise. Suppose that the channel coefficients and the variance of $z_i$ ($\sigma^2$) are known. Let $n_{y_i}$ denote the set of the input variables indexed by $n_i$: $\{x_j : j \in n_i\}$. Then the distribution of $y_i$ conditioned on $n_{y_i}$ is

$$p(y_i|n_{y_i}) = \frac{1}{\pi\sigma^2} \exp\left\{ -\frac{1}{\sigma^2} \left| y_i - \sum_{j \in n_i} h_{i,j} x_j \right|^2 \right\}. \tag{4.17}$$

Suppose that the inputs are independent, then the joint distribution of $\{x_1, \ldots, x_n\}$ is

$$\begin{aligned} g(x_1, \ldots, x_n) &= p(x_1, \ldots, x_n, y_1, \ldots, y_m) \\ &= \prod_{i=1}^{m} p(y_i|n_{y_i}) \prod_{j=1}^{n} p(x_j). \end{aligned} \tag{4.18}$$

We can use the (loopy) factor graph corresponding to the factorization in (4.18) and the sum–product algorithm on that graph to compute (an approximation of) the APP $p(x_i|y_1, \ldots, y_m) \propto g_i(x_i)$. The factor graph corresponding to (4.18) is given in Figure 4.4.

There are two types of messages in Figure 4.4: $x$-to-$y$ messages, and $y$-to-$x$ messages. Let $n_{x_j}$ denote the set of $y_i$ nodes such that $y_i$ is a neighboring factor node of the variable node $x_j$ in the graph. If $y_i \in n_{x_j}$, the message from variable

node $x_j$ to factor node $y_i$ is, from (4.6),

$$\mu_{x_j - y_i}(x_j) = p(x_j) \prod_{y_k \in n_{x_j} \setminus \{y_i\}} \mu_{y_k - x_j}(x_j). \tag{4.19}$$

If $x_j \in n_{y_i}$, the message from factor node $y_i$ to variable node $x_j$ is, from (4.7),

$$\mu_{y_i - x_j}(x_j) = \sum_{\sim\{x_j\}} \left( f(y_i | n_{y_i}) \prod_{x_l \in n_{y_i} \setminus \{x_j\}} \mu_{x_l - y_i}(x_l) \right). \tag{4.20}$$

During initialization, the pendant factor nodes $p(x_j)$ send their description to variable nodes $x_j$. In addition, the factor nodes $p(y_j|\cdot)$ send trivial messages to their neighboring variable nodes: $\mu_{y_i - x_j}(x_j) = 1$ for $i = 1, \ldots, m$, and $x_j \in n_{y_i}$. Afterwards, we have an iterative algorithm, where at each iteration we compute

(1) $x$-to-$y$ messages for each $j \in \{1, \ldots, n\}$ and $y_i \in n_{x_j}$ in (4.19);
(2) $y$-to-$x$ messages for each $i \in \{1, \ldots, m\}$ and $x_j \in n_{y_i}$ in (4.20).

Notice that the graph in Figure 4.4 is loopy, and this means that the algorithm will not terminate in a finite number of steps, nor will it be guaranteed to find the correct marginalizations. If the algorithm does converge, however, then it can be terminated after a sufficiently large number of steps, and then an approximation to the marginal distribution on the variable nodes can be obtained as follows. The messages at variable node $x_j$ for $j \in \{1, \ldots, n\}$ are combined as

$$\mu_{x_j}(x_j) = p(x_j) \prod_{y_k \in n_{x_j}} \mu_{y_k - x_j}(x_j). \tag{4.21}$$

Models that lead to factor graphs with loops like this simple interference channel example will arise when we turn our attention to two-dimensional cellular network models in Section 4.4. First, however, we will look at one-dimensional cellular networks where the corresponding factor graphs are loop-free.

## 4.3 Distributed decoding in the uplink: one-dimensional cellular model

Consider again the cellular network where the BSs and the cells are placed on a line, as depicted in Figure 4.1. In this model, every cell has two neighboring cells. Although this simple model is far from being realistic, it provides a framework in which the main concepts of distributed processing with message passing can be developed and explained, and it can then be generalized to less restrictive models.

Let $x_i$ denote the symbol transmitted by MS $i$ and $y_i$ denote the channel output observed at BS $i$, as depicted in Figure 4.2. In the linear cellular array model, the relationship between the transmitted symbols and the received signals is described by (4.1). As discussed in Section 4.1, the goal is to obtain optimal detection of any particular $x_i$ given all observations $y_1, \ldots, y_n$, in a distributed

manner with cooperating BSs, as an alternative to the traditional approach of SCP. In SCP, BS $i$ has access to the channel output $y_i$ only. In contrast, we are interested here in distributed, message-passing algorithms to accomplish MCP, based on probabilistic graphical models.

### 4.3.1 Hidden Markov model and the factor graph

The linear cellular array model is highly reminiscent of a standard linear ISI model in digital communications, and hence we expect to be able to apply the BCJR algorithm [8]. In [8], a state-based hidden Markov model is used, as described in Example 1 in Section 4.2.2. In a state-based model, several input variables are combined to form a state such that each channel output is only a function of that state, and the state sequence forms a Markov chain.

The key idea in [21] is to treat the one-dimensional cellular model as an ISI channel. In fact, this idea goes back to [59]. The state for cell $i$ is $s_i = (x_{i-1}, x_i, x_{i+1})$ and we assume the symbols from different mobiles are independent, taking values in some finite alphabet (which can be different for the different users). Thus, there are several possible values for the state $s_i$, so we will write $(x_{i-1}(s_i), x_i(s_i), x_{i+1}(s_i))$ for the values of the data symbols corresponding to a particular state value $s_i$. It is clear that the state sequence is a Markov chain, with the following transition probabilities:

$$p(s_1) = p(x_0(s_1))p(x_1(s_1))p(x_2(s_1)),$$
$$p(s_{i+1}|s_i) = [x_i(s_i) = x_i(s_{i+1})][x_{i+1}(s_i) = x_{i+1}(s_{i+1})]p(x_{i+2}(s_{i+1})),$$

where the $[P]$ notation was defined in (4.10). Note that $[x_i(s_i) = x_i(s_{i+1})][x_{i+1}(s_i) = x_{i+1}(s_{i+1})]$ indicates whether state $s_{i+1}$ conforms with state $s_i$, i.e., whether a transition from $s_i$ to $s_{i+1}$ is possible.

Note that each cell has one channel output, $y_i$, which is dependent only on the state $s_i$, as in the hidden Markov model of Section 4.2.2. To complete the match with the model in that section, we define the output variable corresponding to the transition from $s_{i-1}$ to $s_i$ to be $u_i$, where

$$u_i = d(s_i) := h_i(-1)x_{i-1}(s_i) + h_i(0)x_i(s_i) + h_i(+1)x_{i+1}(s_i),$$

and we note that the conditional distribution of the observation, $y_i$, given $u_i$, is $f(y_i|u_i) = N(y_i; \sigma^2, u_i)$, where $N(x; \sigma^2, M)$ denotes the Gaussian distribution with mean $M$ and variance $\sigma^2$. The corresponding factor graph is shown in Figure 4.5, where the function node $T_i$ computes the function:

$$T_i(s_{i-1}, u_i, s_i) = p(s_i|s_{i-1})[u_i = d(s_i)]. \tag{4.22}$$

It follows that the forward–backward algorithm can be applied to obtain the APP $p(s_i|y_1, \ldots, y_n)$, which can be further marginalized to obtain $p(x_i|y_1, \ldots, y_n)$, the APP of the mobile data symbols.

The implementation of the forward–backward recursions is distributed among the BSs, where BS $i$ performs the computations done by node $T_i$ in the algorithm.

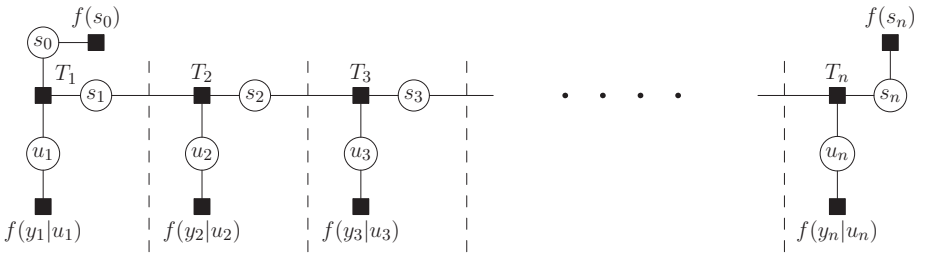



Figure 4.5. Factor graph for the hidden Markov model for the linear cellular array. Dashed lines show boundaries between cells. The computations of the nodes within a cell are done by the BS of that cell. Any message passing through a cell boundary corresponds to actual message passing between corresponding BSs.

For example, upon receiving the message $\alpha(s_{i-1})$ from cell $i-1$, BS $i$ computes $\alpha(s_i)$ and forwards it to cell $i+1$. Thus, $\alpha$ messages ripple across the BSs from left to right, and $\beta$ messages ripple in the reverse direction. After the forward and backward recursion is complete, the APP $p(s_i|y_1,\ldots,y_n)$ is obtained as a scaled version of (4.15). In this formulation, the middle BS is the first to be able to decode its mobile.

This serial formulation of the forward–backward algorithm is the natural one to use in solving an ISI equalization problem. It is not natural, however, in cellular radio networks to designate a leftmost or rightmost BS. In fact, we cannot do that at all for an infinite linear array model. Fortunately, the sum–product algorithm has flexibility in terms of node activation schedules [30]. Initial conditions can be arbitrary, and each node can operate in parallel. This allows all BSs to immediately begin computing their messages starting with the a priori distributions on the input symbols. At each iteration, a BS passes an $\alpha$ message to the right, and a $\beta$ message to the left. In a finite linear array, this parallel version of the forward–backward algorithm converges to the same solution as obtained from the serial implementation, but an important point is that it can be terminated early giving a suboptimal estimate of the mobile's data symbol at an earlier time. In the infinite linear array, the algorithm must be terminated at some point in time. This approach allows an investigation of estimation error versus delay, as can be found in [39].

The actual values that the variables can take have not been specified. In this section, we have in mind that each $x_i$ takes a value from a discrete constellation, and, as such, the BSs are engaged in the demodulation of the users' data symbols. If the symbol $x_i$ is replaced by the transmitted codeword of mobile $i$ and $y_i$ is replaced by the channel outputs corresponding to a codeword, i.e., if we include the time dimension, then we can use the described method for decoding as opposed to the detection of individual symbols, as considered in [21]. In the present section, the forward–backward algorithm is accomplishing joint

multiuser detection (MUD) of the users' data symbols, prior to single-user decoding. After the detection of the symbols, each BS can decode its own user using a single-user decoder.

Note that the complexity of MUD is typically exponential in the number of users [54], but it is known that in some special cases the complexity can be much reduced [47, 52], for example when the signature sequences have constant cross-correlation [48]. In the present section, we have a distributed MUD that is linear in the number of users, and this is due to the highly localized interference model: the cross-correlations of most signature sequences are zero. Indeed, the BCJR algorithm implements the optimal MAP detection of the users' symbols, and this is known to have a complexity that is linear in time [8], i.e., in the number of symbols.

To approach Shannon capacity at high SNR, it is required to send many bits per symbol, which requires a large alphabet size (large signal constellations), and the BCJR algorithm is exponential in the alphabet size. So even if the complexity is linear in the number of users, the overall complexity can be very high. This observation also applies to the decoding of codewords in the model considered in [21]. A standard approach to limit the complexity of MUD is to restrict attention to suboptimal linear techniques, which we consider further in Section 4.3.2. Unfortunately, this does not avoid the complexity of the overall decoding problem, but at least one can then focus attention on well-established techniques for decoding *single-user* codes.

### 4.3.2 Gaussian symbols

A standard approach in MUD is first to estimate the individual symbols from different users using linear MUD techniques. Once the BS has estimated symbol $x_i$ from mobile $i$ it then passes this soft estimate to a single-user decoder for mobile $i$. The decoder waits until it receives the estimates of all symbols in the codeword, and then it attempts to decode the codeword. This approach limits the complexity of the MUD component of the receiver.

It is well known that optimal MUD is in fact linear if the underlying symbols being estimated are jointly Gaussian. In this section, we *assume* that the input symbols are drawn from joint Gaussian distributions (independent across mobiles) and then we apply the corresponding optimal linear filters, and the task of the present section is to show how these filters can be implemented via message passing between the BSs in the cellular network. Another motivation for this section is that the developed methods will prove useful in designing iterative message-passing algorithms to accomplish beamforming on the downlink of a cellular system, as we will see in Section 4.5.

When the input symbols are modeled as Gaussian random variables, we can still employ factor graph methods. The global function is now a continuous function and the marginalization is done by integrating (as opposed to summing) with respect to unwanted variables. Since the messages are now continuous

functions, each message in general corresponds to a continuum of values. However, if the message functions can be parameterized, they can be represented by a finite number of parameter values. For example, if a message function is the probability density function of a Gaussian vector, then it is characterized by a mean vector and covariance matrix pair, which is the case for the Gaussian input model.

We will now describe the Kalman-smoothing-based distributed algorithm in [39] for the linear cellular array. The model is the same as in (4.1) except that now the $x_i$s are independent zero-mean Gaussian distributed with variance $p$. We are going to use matrix-vector notation, so define the state for cell $i$ to be the column vector $\mathbf{s}_i = [x_{i-1}, x_i, x_{i+1}]^T$. The states again form a Markov chain, but we now express the transition from state $\mathbf{s}_i$ to $\mathbf{s}_{i+1}$ as

$$\mathbf{s}_{i+1} = \mathbf{A}^f \mathbf{s}_i + \mathbf{b}^f x_{i+2},$$

where

$$\mathbf{A}^f = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{b}^f = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Then the state transition is characterized by the conditional distribution

$$f(\mathbf{s}_{i+1}|\mathbf{s}_i) = N(\mathbf{s}_{i+1}; p\mathbf{b}^f \mathbf{b}^{f\,T}, \mathbf{A}^f \mathbf{s}_i), \tag{4.23}$$

where we use the notation

$$N(\mathbf{s}; \mathbf{M}, \mathbf{m}) \propto \exp\left\{-\frac{1}{2}(\mathbf{s}-\mathbf{m})^T \mathbf{M}^{-1}(\mathbf{s}-\mathbf{m})\right\}$$

to denote a Gaussian distribution, scaled by an arbitrary constant that is not a function of the argument of the function. Here, $\mathbf{s}$ is the argument of the function and $\mathbf{M}$ and $\mathbf{m}$ are parameters.

Define the column vector

$$\mathbf{h}_i = \begin{bmatrix} h_i(-1) & h_i(0) & h_i(1) \end{bmatrix}^T,$$

then the observation in cell $i$ can be expressed in vector form as

$$y_i = \mathbf{h}_i^T \mathbf{s}_i + z_i.$$

The factorization of the joint distribution again has the form in (4.12). The corresponding factor graph is shown in Figure 4.6.

Since the $\mathbf{s}_i$s and $y_i$s are jointly Gaussian, all of the messages turn out to be Gaussian distributions. Thus the actual messages will be the mean vector and covariance matrix pairs.

Before deriving the messages, let us present some useful results for the Gaussian distribution. Remember that in our notation the distribution is scaled by

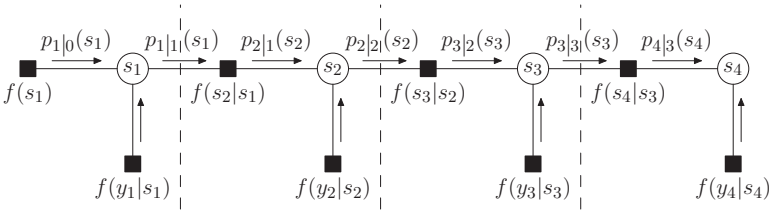



Figure 4.6. Factor graph for a hidden Markov model for $n = 4$ used for the linear cellular array with Gaussian inputs. Dashed lines show boundaries between cells. The computations of the nodes within a cell are done by the base station of that cell. Any message passing through a cell boundary corresponds to an actual message passing between corresponding base stations.

an arbitrary constant.

$$N(\mathbf{s}; \mathbf{M}, \mathbf{m}) = N(\mathbf{m}; \mathbf{M}, \mathbf{s}), \tag{4.24}$$

$$N(\mathbf{As} + \mathbf{b}; \mathbf{M}, \mathbf{m}) = N(\mathbf{s}; \mathbf{A}^{-1}\mathbf{M}\mathbf{A}^{-1^T}, \mathbf{A}^{-1}(\mathbf{m} - \mathbf{s})), \tag{4.25}$$

$$N(\mathbf{s}; \mathbf{M}_1, \mathbf{m}_1)N(\mathbf{s}; \mathbf{M}_2, \mathbf{m}_2) = N(\mathbf{s}; \mathbf{M}_3, \mathbf{m}_3),$$

where

$$\mathbf{M}_3 = (\mathbf{M}_1^{-1} + \mathbf{M}_2^{-1})^{-1}, \quad \mathbf{m}_3 = \mathbf{M}_3(\mathbf{M}_1^{-1}\mathbf{m}_1 + \mathbf{M}_2^{-1}\mathbf{m}_2); \tag{4.26}$$

$$N(\mathbf{s}; \mathbf{M}_1, \mathbf{m}_1)N(\mathbf{s}; \mathbf{M}_2, \mathbf{m}_2)^{-1} = N(\mathbf{s}; \mathbf{M}_4, \mathbf{m}_4),$$

where

$$\mathbf{M}_4 = (\mathbf{M}_1^{-1} - \mathbf{M}_2^{-1})^{-1}, \quad \mathbf{m}_4 = \mathbf{M}_4(\mathbf{M}_1^{-1}\mathbf{m}_1 - \mathbf{M}_2^{-1}\mathbf{m}_2); \tag{4.27}$$

$$\int N(\mathbf{s}; \mathbf{M}_1, \mathbf{m}_1)N(\mathbf{As}; \mathbf{M}_2, \mathbf{t})\, d\mathbf{s} = N(\mathbf{t}; \mathbf{A}\mathbf{M}_1\mathbf{A}^T + \mathbf{M}_2, \mathbf{A}\mathbf{m}_1). \tag{4.28}$$

We know that the messages are going to be Gaussian. Denote them by

$$p_{i|i-1}(\mathbf{s}_i) = N(\mathbf{s}_i; \mathbf{M}_{i|i-1}, \hat{\mathbf{s}}_{i|i-1}), \tag{4.29}$$

$$p_{i|i}(\mathbf{s}_i) = N(\mathbf{s}_i; \mathbf{M}_{i|i}, \hat{\mathbf{s}}_{i|i}). \tag{4.30}$$

From the observation node, we have the message

$$f(y_i|\mathbf{s}_i) = N(y_i; \sigma^2, \mathbf{h}_i^T \mathbf{s}_i).$$

From (4.6), the message from variable node $\mathbf{s}_i$ to factor node $f(\mathbf{s}_{i+1}|\mathbf{s}_i)$ is

$$\begin{aligned} p_{i|i}(\mathbf{s}_i) &= p_{i|i-1}(\mathbf{s}_i) f(y_i|\mathbf{s}_i) \\ &\propto \exp\left\{-\frac{1}{2}\left[(\mathbf{s}_i - \hat{\mathbf{s}}_{i|i-1})^T \mathbf{M}_{i|i-1}^{-1}(\mathbf{s}_i - \hat{\mathbf{s}}_{i|i-1})\right]\right\} \exp\left\{-\frac{1}{2}\left[\frac{1}{\sigma^2}(y_i - \mathbf{h}_i^T \mathbf{s}_i)^2\right]\right\} \\ &\propto \exp\left\{-\frac{1}{2}\left[(\mathbf{s}_i - \hat{\mathbf{s}}_{i|i})^T \mathbf{M}_{i|i}^{-1}(\mathbf{s}_i - \hat{\mathbf{s}}_{i|i})\right]\right\}, \end{aligned}$$

where

$$\mathbf{M}_{i|i} = \left(\mathbf{M}_{i|i-1}^{-1} + \frac{1}{\sigma^2}\mathbf{h}_i\mathbf{h}_i^T\right)^{-1}, \tag{4.31}$$

$$\hat{\mathbf{s}}_{i|i} = \mathbf{M}_{i|i}\left(\mathbf{M}_{i|i-1}^{-1}\hat{\mathbf{s}}_{i|i-1} + \frac{1}{\sigma^2}\mathbf{h}_i y_i\right). \tag{4.32}$$

Thus the pair (4.31)–(4.32) is the message from $\mathbf{s}_i$ to $f(s_{i+1}|s_i)$. This pair of equations is another form of the more familiar Kalman filter correction update [27]:

$$\mathbf{M}_{i|i} = \left(\mathbf{I} - \mathbf{K}_i\mathbf{h}_i^T\right)\mathbf{M}_{i|i-1}, \tag{4.33}$$

$$\hat{\mathbf{s}}_{i|i} = \hat{\mathbf{s}}_{i|i-1} + \mathbf{K}_i\left(y_i - \mathbf{h}_i^T\hat{\mathbf{s}}_{i|i-1}\right), \tag{4.34}$$

where

$$\mathbf{K}_i = \frac{\mathbf{M}_{i|i-1}\mathbf{h}_i}{\sigma^2 + \mathbf{h}_i^T\mathbf{M}_{i|i-1}\mathbf{h}_i}.$$

The equivalence of (4.31)–(4.32) and (4.33)–(4.34) can be shown using inversion of matrix sum identities.

Next, let us obtain the message function $p_{i|i-1}(\mathbf{s}_i)$ using (4.7):

$$p_{i|i-1}(\mathbf{s}_i) = \int f(\mathbf{s}_i|\mathbf{s}_{i-1})p_{i-1|i-1}(\mathbf{s}_{i-1})\,d\mathbf{s}_{i-1}. \tag{4.35}$$

Note that the summation in (4.7) becomes integration in (4.35) since we are dealing with continuous variables. From (4.23) and (4.30):

$$\begin{aligned} p_{i|i-1}(\mathbf{s}_i) &= \int N(\mathbf{s}_i; p\mathbf{b}^f\mathbf{b}^{fT}, \mathbf{A}^f\mathbf{s}_{i-1})N(\mathbf{s}_{i-1}, \mathbf{M}_{i-1|i-1}, \hat{\mathbf{s}}_{i-1|i-1})\,d\mathbf{s}_{i-1} \\ &\propto N(\mathbf{s}_i; p\mathbf{b}^f\mathbf{b}^{fT} + \mathbf{A}^f\mathbf{M}_{i-1|i-1}\mathbf{A}^{fT}, \mathbf{A}^f\hat{\mathbf{s}}_{i-1|i-1}), \end{aligned} \tag{4.36}$$

where (4.36) is due to (4.24) and (4.28). As a result, the message function $p_{i|i-1}(\mathbf{s}_i)$ is represented by the mean-covariance pair:

$$\hat{\mathbf{s}}_{i|i-1} = \mathbf{A}^f\hat{\mathbf{s}}_{i-1|i-1}, \tag{4.37}$$

$$\mathbf{M}_{i|i-1} = p\mathbf{b}^f\mathbf{b}^{fT} + \mathbf{A}^f\mathbf{M}_{i-1|i-1}\mathbf{A}^{fT}. \tag{4.38}$$

Equations (4.37)–(4.38) are Kalman filter prediction updates [27].

Note that the message $p_{i|i}(\mathbf{s}_i)$ is the posterior distribution of $\mathbf{s}_i$ given $\{y_1, \ldots, y_i\}$, and $p_{i|i-1}(\mathbf{s}_i) = f(\mathbf{s}_i|y_1, \ldots, y_{i-1})$. We desire the posterior distribution of $\mathbf{s}_i$ given all observations: $f(\mathbf{s}_i|y_1, \ldots, y_n)$. For that purpose, form a graph similar to Figure 4.6 but in the backward direction: states are ordered from $\mathbf{s}_N$ to $\mathbf{s}_1$ and connected by the transition nodes $f(\mathbf{s}_{i-1}|\mathbf{s}_i)$, where [39]

$$f(\mathbf{s}_{i-1}|\mathbf{s}_i) = N(\mathbf{s}_{i-1}; p\mathbf{b}^b\mathbf{b}^{bT}, \mathbf{A}^b\mathbf{s}_i),$$

$$\mathbf{A}^b = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad \mathbf{b}^b = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

For the backward graph, denote the message from factor node $f(\mathbf{s}_i|\mathbf{s}_{i+1})$ to variable node $\mathbf{s}_i$ by

$$p_{i|i+1}(\mathbf{s}_i) = N(\mathbf{s}_i; \mathbf{M}_{i|i+1}, \hat{\mathbf{s}}_{i|i+1}),$$

which will be the posterior distribution of $\mathbf{s}_i$ given $\{y_i + 1, \ldots, y_n\}$. Combination of the backward message $p_{i|i+1}(\mathbf{s}_i)$ with the forward message $p_{i|i}(\mathbf{s}_i)$ to obtain $f(s_i|y_1, \ldots, y_n)$ can be done as follows:

$$\begin{aligned}
f(\mathbf{s}_i|y_1, \ldots, y_n) &\propto f(\mathbf{s}_i, y_1, \ldots, y_n) \\
&= f(y_1, \ldots, y_i|\mathbf{s}_i, y_{i+1}, \ldots, y_n) f(\mathbf{s}_i, y_{i+1}, \ldots, y_n) \\
&\propto f(y_1, \ldots, y_i|\mathbf{s}_i) f(\mathbf{s}_i|y_{i+1}, \ldots, y_n) \qquad (4.39) \\
&\propto f(\mathbf{s}_i|y_1, \ldots, y_i) f(\mathbf{s}_i|y_{i+1}, \ldots, y_n) f(\mathbf{s}_i)^{-1} \\
&= p_{i|i}(\mathbf{s}_i) p_{i|i+1}(\mathbf{s}_i) f(\mathbf{s}_i)^{-1} \\
&= N(\mathbf{s}_i; \mathbf{M}_{i|i}, \hat{\mathbf{s}}_{i|i}) N(\mathbf{s}_i; \mathbf{M}_{i|i+1}, \hat{\mathbf{s}}_{i|i+1}) N(\mathbf{s}_i; p\mathbf{I}, \mathbf{0})^{-1} \qquad (4.40) \\
&= N(\mathbf{s}_i; \mathbf{M}_3, \mathbf{m}_3) N(\mathbf{s}_i; p\mathbf{I}, \mathbf{0})^{-1} \qquad (4.41) \\
&= N(\mathbf{s}_i; \mathbf{M}_i, \hat{\mathbf{s}}_i). \qquad (4.42)
\end{aligned}$$

Equation (4.39) is due to the fact that given $\mathbf{s}_i$, $\{y_1, \ldots, y_i\}$ and $\{y_{i+1}, \ldots, y_n\}$ become independent. In (4.40) the fact that the prior distribution of $\mathbf{s}_i$ is zero-mean Gaussian with covariance $p\mathbf{I}$ is used. Equation (4.41) is from (4.26), where

$$\mathbf{M}_3 = (\mathbf{M}_{i|i}^{-1} + \mathbf{M}_{i|i+1}^{-1})^{-1}, \qquad (4.43)$$

$$\mathbf{m}_3 = \mathbf{M}_3(\mathbf{M}_{i|i}^{-1}\hat{\mathbf{s}}_{i|i} + \mathbf{M}_{i|i+1}^{-1}\hat{\mathbf{s}}_{i|i+1}). \qquad (4.44)$$

Equation (4.42) is from (4.27), where

$$\mathbf{M}_i = (\mathbf{M}_3^{-1} - p^{-1}\mathbf{I})^{-1}, \qquad (4.45)$$

$$\hat{\mathbf{s}}_i = \mathbf{M}_i(\mathbf{M}_3^{-1}\mathbf{m}_3). \qquad (4.46)$$

Combining (4.43)–(4.46), we obtain the result

$$\begin{aligned}
\mathbf{M}_i &= \left(\mathbf{M}_{i|i}^{-1} + \mathbf{M}_{i|i+1}^{-1} - \frac{1}{p}\mathbf{I}\right)^{-1}, \\
\hat{\mathbf{s}}_i &= \mathbf{M}_i(\mathbf{M}_{i|i}^{-1}\hat{\mathbf{s}}_{i|i} + \mathbf{M}_{i|i+1}^{-1}\hat{\mathbf{s}}_{i|i+1}).
\end{aligned}$$

For the one-dimensional cellular network we have seen how message passing algorithms can be applied on the uplink to detect discrete data symbols and estimate Gaussian data symbols. The one-dimensional nature of these models leads to underlying factor graphs without loops and thus to guaranteed convergence of the sumproduct algorithm on these factor graphs. In the sequel, we will see that the situation is quite different when we move to two-dimensional cellular networks.

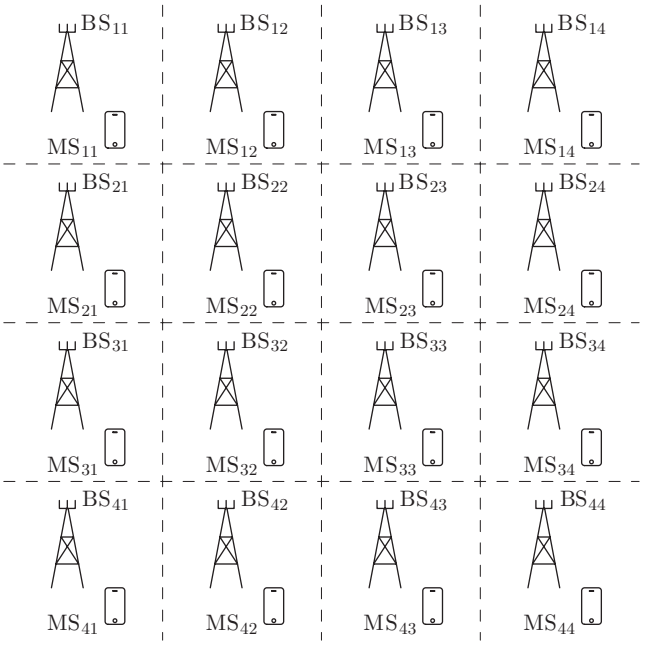



Figure 4.7. Rectangular cellular array model. The cells are positioned on a rectangular grid. Each cell has one active MS. The signal transmitted in one cell is received at that cell, and also four neighboring cells (except for edge cells). Dashed lines show boundaries between cells.

## 4.4 Distributed decoding in the uplink: two-dimensional cellular array model

### 4.4.1 The rectangular model

A model that is more general than the linear array model is the model where BSs are positioned on a two-dimensional grid. For example, consider the rectangular model where the BSs are on a rectangular grid. Again, assume flat fading and orthogonal multiple access channels within a cell. The received signal at the BS of any cell, in any channel, is the superposition of the signal from its own MS, and the signals of the four adjacent cell cochannel users. The positioning of the BSs and MSs is shown in Figure 4.7.

For cell $(i, j)$ in the rectangular grid, let $x_{i,j}$ denote the symbol transmitted by the MS, $y_{i,j}$ the signal received at the BS, $h_{i,j}(x_{m,n})$ the channel from mobile $(m, n)$ to BS $(i, j)$, and $z_{i,j}$ additive Gaussian noise. The relationship between the observations $y_{i,j}$ and the transmitted symbols $x_{i,j}$ is expressed as

$$y_{i,j} = \sum_{x_{m,n} \in n_{y_{i,j}}} h_{i,j}(x_{m,n}) x_{m,n} + z_{i,j}, \tag{4.47}$$

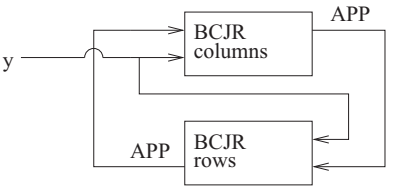



Figure 4.8. Iterative implementation of BCJR algorithm along the columns and rows of the rectangular array (© 2008 IEEE).

where

$$n_{y_{i,j}} = \{x_{i,j}, x_{i-1,j}, x_{i+1,j}, x_{i,j-1}, x_{i,j+1}\} \tag{4.48}$$

is the set of transmitted symbols that can be heard at BS $(i, j)$. For the cells at the edges of the rectangular network, dummy symbols $x_{0,j}$, $x_{n+1,j}$, $x_{i,0}$, $x_{i,n+1}$ are added and the corresponding $h_{i,j}(x_{m,n})$ are set to zero.

The goal is again to obtain the global APP $p(x_{i,j}|y)$, where $y = \{y_{1,1}, \ldots, y_{1,n}, \ldots, y_{n,1}, \ldots, y_{n,n}\}$ is the set of all observations. It is still possible to obtain exact inference by forming a Markov chain via clustering (e.g., states obtained by clustering along the rows of the two-dimensional array) and then apply the BCJR algorithm, but the complexity grows exponentially with $n$ (the number of columns or rows in the rectangular array) and is intractable as the network size grows. It is possible that the inherent complexity is only polynomial in the network size (we have not investigated this issue) but in any case, we are looking for distributed approaches using message passing between neighboring base stations.

Encouraged by the elegance of the implementation of the BCJR algorithm for the one-dimensional array, one can be tempted to use this approach along the columns and rows of a rectangular array in an iterative manner. The APP outputs of the BCJR along one direction will be used as a priori probabilities for the BCJR along the other direction. Thus the global decoder is built as an iterative decoder where the two modules of the iterative decoder are the BCJR in each direction (Figure 4.8 from [5]).

The details of this approach, and a discussion of its implementation are in [4]. The idea of running BCJR along the rows and columns of a rectangular cellular array was also proposed for two-dimensional ISI channels by Marrow and Wolf in [35].

Although applying the BCJR algorithm along the rows and columns of the rectangular array seems to work [5, 35], we see that it does not directly exploit the two-dimensional structure of the problem but instead imposes a one-dimensional structure on parts of it. As this is an ad-hoc iterative method, it will result in only an approximation to the desired APPs. However, if we are looking for an approximation of the APP, there is no need to impose the use of the BCJR algorithm which gives the optimum result only if the problem is one-dimensional.

Thus we can accept an approximate APP and form loopy graphs that reflect the true two-dimensional nature of the problem.

### 4.4.2 Earlier methods not based on graphs

Research has considered distributed global demodulation in two-dimensional cellular channels [53, 57]. In [53], the authors considered BSs computing soft estimates of the symbols, and then sharing and combining them to obtain a final soft estimate. This strategy was compared with BSs sharing channel outputs and performing maximum-ratio combining of the channel outputs. In [57], a reduced complexity maximum-likelihood (ML) decoder was developed, which was motivated as an extension of the Viterbi algorithm which exploits the limited interference structure. Although the general large two-dimensional cellular structure was not treated, it seems that the algorithm, if applied to that structure, would result in increasing complexity per symbol with growing network size.

Alternatively, graph-based iterative message passing methods for distributed detection for two-dimensional cellular networks were proposed in [3–5, 50].

### 4.4.3 State-based graph approach

One way to model the two-dimensional case is to adapt the state-based graphical idea from the one-dimensional case. Remember that in a state-based graph, each channel output depends only on one state variable, and everything else in the system is modeled by the transitions among the states. For the one-dimensional case, the states form a Markov chain, but in the two-dimensional case they do not.

For cell $(i, j)$, define the state to be

$$s_{i,j} = n_{y_{i,j}}, \tag{4.49}$$

where $n_{y_{i,j}}$ is the set of symbols defined in (4.48), upon which $y_{i,j}$ depends, as in

$$y_{i,j} = m_{i,j}(s_{i,j}) + z_{i,j},$$

where the conditional mean $m_{i,j}(s_{i,j})$ is a deterministic function of $s_{i,j}$,

$$m_{i,j}(s_{i,j}) = \sum_{x_{m,n} \in n_{y_{i,j}}} h_{i,j}(x_{m,n}) x_{m,n}. \tag{4.50}$$

It can be observed that the states form a Markov random field, a fact that we will exploit in (4.51).

As in the one-dimensional model, the variables in the sum–product algorithm are not the channel input symbols $x_{i,j}$, but the states $s_{i,j}$. The goal of BS $(i, j)$ is to obtain the APP $p(s_{i,j}|y)$, from the marginalization of which it can obtain the APP $p(x_{i,j}|y)$. Let $s = \{s_{i,j}\}$ denote the set of all states. The global function

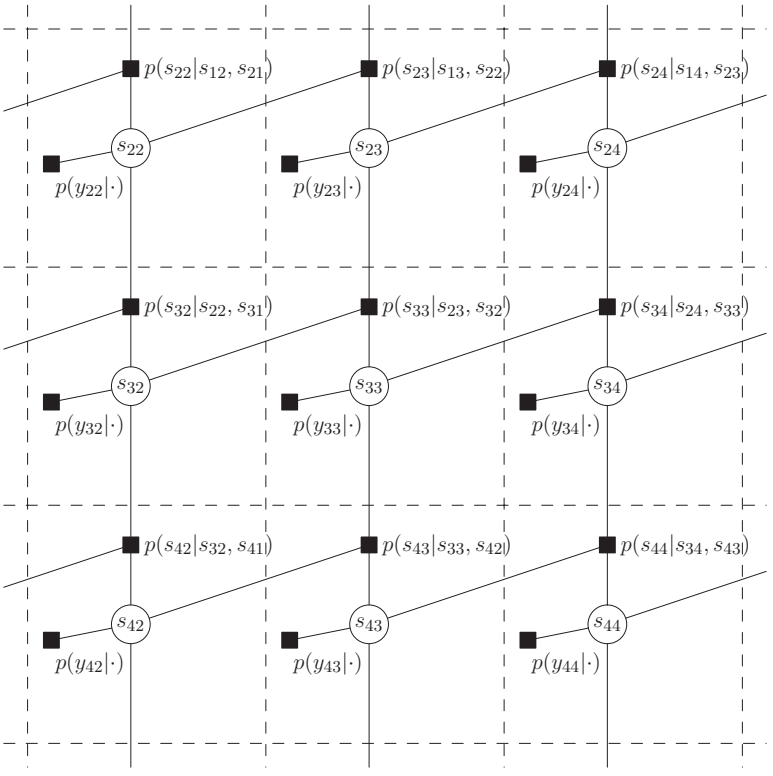



Figure 4.9. Factor graph for the state-based probabilistic model for the rectangular cellular array. Dashed lines show boundaries between cells. The computations of the nodes within a cell are done by the BS of that cell. Any message passing through a cell boundary corresponds to actual message passing between corresponding BSs.

to be marginalized to compute the $p(s_{i,j}|y) \propto g_i(s_i)$ is

$$\begin{aligned} g(s) &= p(y, s) \\ &= \prod_{i=1}^{n} \prod_{j=1}^{n} p(y_{i,j}|s_{i,j}) p(s_{i,j}|s_{i-1,j}, s_{i,j-1}), \end{aligned} \tag{4.51}$$

where $s_{0,j}$ and $s_{i,0}$ are dummy states: $p(s_{1,j}|s_{0,j}, s_{1,j-1}) = p(s_{1,j}|s_{1,j-1})$ and $p(s_{i,1}|s_{i-1,1}, s_{i,0}) = p(s_{i,1}|s_{i-1,1})$.

The factor graph for the factorization in (4.51) is depicted in Figure 4.9. Note that this is a loopy graph and hence the sum–product algorithm is not guaranteed to give the exact APP. Nevertheless, we now describe the algorithm which will be observed to provide good performance in practical settings. Uniform prior distributions $p(x_{i,j})$ are assumed on the input symbol constellations. Define the

set of states

$$p_{i,j} = \{p_{i,j}^1, p_{i,j}^2\},$$

where

$$\begin{aligned} p_{i,j}^1 &= s_{i-1,j}, \\ p_{i,j}^2 &= s_{i,j-1}. \end{aligned}$$

The system is modeled by transitions from $p_{i,j}$ to $s_{i,j}$. Note that not every transition $p_{i,j} \to s_{i,j}$ is possible. Similarly to the one-dimensional case, we say that $s_{i,j}$ is *conformable* with $p_{i,j}$ if there is transition from $p_{i,j}$ to $s_{i,j}$, i.e., $p(s_{i,j}|p_{i,j}) > 0$ for some prior distribution on the $x_{i,j}$s. The set of all configurations of $p_{i,j}$ that are conformable with $s_{i,j}$ will be denoted by $p_{i,j} : s_{i,j}$, and the set of all configurations of $s_{i,j}$ that are conformable with $p_{i,j}$ will be denoted by $s_{i,j} : p_{i,j}$.

The message computations described next for cell $(i, j)$ can be implemented at BS $(i, j)$, simultaneously in parallel by all BSs. The message from $p(y_{i,j}|\cdot)$ to $s_{i,j}$ is

$$\begin{aligned} \mu_{y_{i,j}-s_{i,j}}(s_{i,j}) &= p(y_{i,j}|s_{i,j}) \\ &= CN(y_{i,j}; m(s_{i,j}), \sigma^2), \end{aligned}$$

where $CN(y; m, \sigma^2)$ is the distribution function of the complex Gaussian with mean $m$ and variance $\sigma^2$. This message is computed only once given an observation $y_{i,j}$ at BS $(i, j)$.

Next, for convenience, define the following factor nodes corresponding to the conditional distribution functions in the factorization in (4.51):

$c_{i,j}^1$ : factor node $p(s_{i,j+1}|s_{i-1,j+1}, s_{i,j})$;
$c_{i,j}^2$ : factor node $p(s_{i+1,j}|s_{i,j}, s_{i+1,j-1})$;
$d_{i,j}$ : factor node $p(s_{i,j}|s_{i-1,j}, s_{i,j-1})$.

The messages between nodes $s_{i,j}$ and $d_{i,j}$ are internal calculations in BS $(i, j)$. The message from factor node $d_{i,j}$ to variable node $s_{i,j}$, from (4.7), is

$$\begin{aligned} \mu_{d_{i,j}-s_{i,j}}(s_{i,j}) &= \sum_{\sim s_{i,j}} p(s_{i,j}|s_{i-1,j}, s_{i,j-1}) \prod_{k=1}^{2} \mu_{p_{i,j}^k - d_{i,j}}(p_{i,j}^k) \\ &= \sum_{p_{i,j}} p(s_{i,j}|p_{i,j}) \prod_{k=1}^{2} \mu_{p_{i,j}^k - d_{i,j}}(p_{i,j}^k) \\ &\propto \sum_{p_{i,j}:s_{i,j}} \prod_{k=1}^{2} \mu_{p_{i,j}^k - d_{i,j}}(p_{i,j}^k) \end{aligned} \tag{4.52}$$

$$\simeq \sum_{\substack{p_{i,j}^1 : s_{i,j} \\ p_{i,j}^2 : s_{i,j}}} \prod_{k=1}^{2} \mu_{p_{i,j}^k - d_{i,j}}(p_{i,j}^k) \tag{4.53}$$

$$= \prod_{k=1}^{2} \bar{\mu}_{p_{i,j}^k - d_{i,j}}(s_{i,j}^k), \tag{4.54}$$

where (4.52) is because $p(s_{i,j}|p_{i,j})$ is a constant if $p_{i,j}$ is conformable with $s_{i,j}$, as prior distribution of $x_{i,j}$s is uniform, and in (4.54)

$$\bar{\mu}_{p_{i,j}^k - d_{i,j}}(s_{i,j}^k) = \sum_{p_{i,j}^k : s_{i,j}} \mu_{p_{i,j}^k - d_{i,j}}(p_{i,j}^k)$$

can be considered as a preprocessed message from $p_{i,j}^k$ to $d_{i,j}$. Note that (4.52) and (4.53) are not, in general, equal because in (4.52) the summation is over $p_{i,j}^1$ and $p_{i,j}^2$, which conform with each other as well as with $s_{i,j}$, whereas in (4.53) we also include $p_{i,j}^1$ and $p_{i,j}^2$, which do not conform with each other. Specifically, $x_{i-1,j-1}$s in $p_{i,j}^1$ and $p_{i,j}^2$ should be the same for the summation in (4.52), but they may be different in (4.53). The additional terms in (4.53) lead to the simplification in (4.54), which results in a considerable saving in complexity. The message from variable node $s_{i,j}$ to factor node $d_{i,j}$, from (4.6), is

$$\mu_{s_{i,j} - d_{i,j}}(s_{i,j}) = \mu_{y_{i,j} - s_{i,j}} \prod_{k=1}^{2} \mu_{c_{i,j}^k - s_{i,j}}(s_{i,j}). \tag{4.55}$$

The message from variable node $s_{i,j}$ to factor node $c_{i,j}^k$ for $k = 1, 2$ is

$$\mu_{s_{i,j} - c_{i,j}^m}(s_{i,j}) = \mu_{d_{i,j} - s_{i,j}} \mu_{y_{i,j} - s_{i,j}} \mu_{c_{i,j}^l - s_{i,j}}(s_{i,j}), \tag{4.56}$$

where $l = \{1, 2\} \backslash k$. The message in (4.56) is an actual message from BS $(i, j)$ to the corresponding BS. Finally, we need the message from $d_{i,j}$ to $p_{i,j}^k$ for $k = 1, 2$, which also should be implemented as an actual message from BS $(i, j)$ to adjacent cells. Using (4.7),

$$\mu_{d_{i,j} - p_{i,j}^k}(p_{i,j}^k) = \sum_{\sim p_{i,j}^k} p(s_{i,j}|p_{i,j}^k, s_{i,j}^l) \mu_{s_{i,j} - d_{i,j}}(s_{i,j}) \mu_{p_{i,j}^l - d_{i,j}}(p_{i,j}^l)$$

$$= \sum_{s_{i,j}} \mu_{s_{i,j} - d_{i,j}}(s_{i,j}) \sum_{p_{i,j}^l} p(s_{i,j}|p_{i,j}^k, p_{i,j}^l) \mu_{p_{i,j}^l - d_{i,j}}(p_{i,j}^l) \tag{4.57}$$

$$\propto \sum_{s_{i,j} : p_{i,j}^k} \mu_{s_{i,j} - d_{i,j}}(s_{i,j}) \sum_{\substack{p_{i,j}^l : s_{i,j} \\ p_{i,j}^l : p_{i,j}^k}} \mu_{p_{i,j}^l - d_{i,j}}(p_{i,j}^l) \tag{4.58}$$

$$\simeq \alpha \sum_{s_{i,j} : p_{i,j}^k} \mu_{s_{i,j} - d_{i,j}}(s_{i,j}) \bar{\mu}_{p_{i,j}^l - d_{i,j}}(s_{i,j}), \tag{4.59}$$

where $l = \{1, 2\} \backslash k$, (4.58) is due to the fact that $p(s_{i,j}|p_{i,j}^k, p_{i,j}^l)$ is a constant for all $s_{i,j} : p_{i,j}^k$ and $p_{i,j}^l : s_{i,j}$. The approximation in (4.59) is similar to (4.53)

in that we have extra terms in the summation with nonconforming $p_{i,j}^k$ and $p_{i,j}^l$. Combining everything, the approximation (due to loops) of the marginal function $g_i(s_i)$ is, from (4.8),

$$\mu_{s_i}(s_i) = \mu_{y_{i,j}-s_{i,j}}(s_{i,j})\mu_{d_{i,j}-s_{i,j}}(s_{i,j})\mu_{c_{i,j}^1-s_{i,j}}(s_{i,j})\mu_{c_{i,j}^2-s_{i,j}}(s_{i,j}). \tag{4.60}$$

For some related work on state-based method for rectangular grids the reader is referred to [50].

### 4.4.4 Decomposed graph approach

As an alternative to the state-based graph approach, let us remember the signal model in (4.47). It is seen that this model is the same as (4.16) in Example 2 of Section 4.2.2. Therefore, for each observation $y_{i,j}$, the conditional distribution given the contributing symbols has the same form as (4.17). As the input symbols are independent, the joint distribution of all symbols and observations has the same form as (4.18). Thus, a factor graph, in the form of Figure 4.4 can be obtained for the purpose of obtaining APPs of the symbols. For the rectangular array model, corresponding to cell $(i, j)$, there will be variable node $x_{i,j}$ and factor node $y_{i,j}$ in this graph, and each factor node $y_{i,j}$ will be connected to the variable nodes in the neighborhood: $n_{y_{i,j}}$ defined as in (4.48). Such a graph for the rectangular model in Figure 4.7 is depicted in Figure 4.10.

Having the factor graph in Figure 4.10, we can perform message passing as described in Example 2 in Section 4.2.2 in order to obtain estimates of the transmitted symbols (the graph clearly has loops). Equations (4.19) and (4.20) are the $x$-to-$y$ and $y$-to-$x$ messages, respectively, except that now we have two indices for each variable, denoting the two-dimensional location of the cell. Any time a message is passed along an edge that crosses a dashed line in Figure 4.10, an actual message passing among corresponding BS is required. At termination, the posterior probability of the transmitted symbol $x_{i,j}$ at cell $(i, j)$ is computed by combining all incoming messages, using (4.21).

In addition to its conceptual simplicity, the decomposed graph approach has the advantage that it does not require the regular positioning of the cells. The method can be applied to any irregular network shape, where each cell has an arbitrary number of neighbors in arbitrary directions.

### 4.4.5 Convergence issues: a Gaussian modeling approach

Unlike in the one-dimensional cellular models, where the graphs are trees, we cannot provide definitive convergence results for our two-dimensional cellular models, in general. It is well known that the sum–product algorithm is not guaranteed to converge when there are loops in the graph. This is the Achilles' heel of our approach to BS cooperation, and it is an area that requires much further study. However, some insights into the convergence properties can be obtained from the Gaussian model, which we consider next.

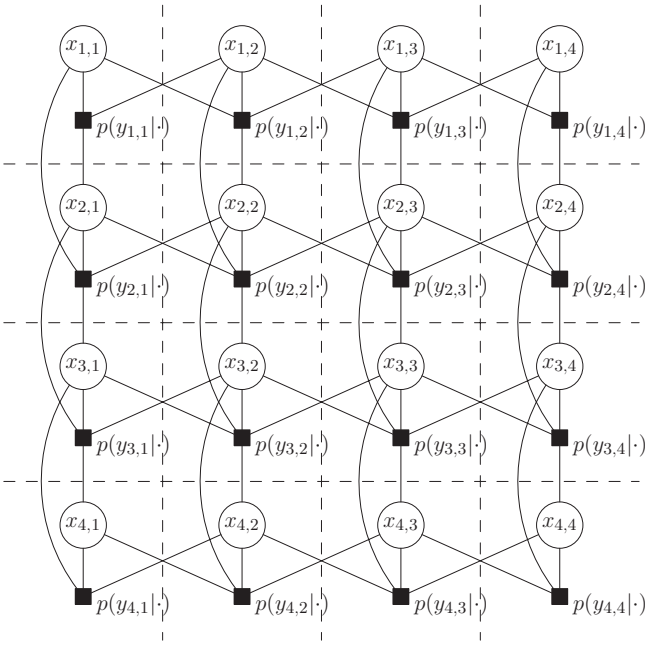



Figure 4.10. Factor graph for the decomposed probabilistic model for the rectangular cellular array model. Dashed lines show boundaries between cells. The computations of the nodes within a cell are done by the BS of that cell. Any message passing through a cell boundary corresponds to actual message passing between corresponding base stations.

In Example 2 in Section 4.2.2, modeling the source symbols $x_j$ as circularly symmetric complex Gaussian in (4.16) also leads to a tractable solution. In that case, all $y_i$s and $x_j$s are jointly Gaussian, and also every local function in the factorization in (4.18) is a Gaussian distribution. As a result, the messages on the graph in Figure 4.10 will also be Gaussian.

Let the pair $(\mu_{x_j-y_i}, \sigma_{x_j-y_i})$ denote the mean-variance pair of the message from $x_j$ to $y_i$, and $(\mu_{y_i-x_j}, \sigma_{y_i-x_j})$ denote the mean-variance pair of the message from $y_i$ to $x_j$. Note that they are both means and variances of the variable $x_j$. Using the properties of complex Gaussian distributions, the mean and variance of the message from $y_i$ to $x_j$ can be shown to be

$$\mu_{y_i-x_j} = \frac{y_i - \sum_{x_l \in n_{y_i} \setminus \{x_j\}} h_i(x_l)\mu_{x_l-y_i}}{h_i(x_j)}, \tag{4.61}$$

$$\sigma_{y_i-x_j} = \frac{\sigma^2 + \sum_{x_l \in n_{y_i} \setminus \{x_j\}} |h_i(x_l)|^2 \sigma_{x_l-y_i}}{|h_i(x_j)|^2}. \tag{4.62}$$

The mean and variance of the message from $x_j$ to $y_i$ can be shown to be

$$\mu_{x_j - y_i} = \frac{\dfrac{\mu^{\pi}_{x_j}}{\sigma^{\pi}_{x_j}} + \displaystyle\sum_{k \in n_{x_j} \setminus \{i\}} \frac{\mu_{y_k - x_j}}{\sigma_{y_k - x_j}}}{\dfrac{1}{\sigma_{x_j}} + \displaystyle\sum_{k \in n_{x_j} \setminus \{i\}} \frac{1}{\sigma_{y_k - x_j}}}, \tag{4.63}$$

$$\sigma_{x_j - y_i} = \left( \frac{1}{\sigma^{\pi}_{x_j}} + \sum_{k \in n_{x_j} \setminus \{i\}} \frac{1}{\sigma_{y_k - x_j}} \right)^{-1}, \tag{4.64}$$

where $(\mu^{\pi}_{x_j}, \sigma^{\pi}_{x_j})$ denotes the prior mean and variance of the symbol $x_j$. At termination, estimates of the mean and variance of the posterior distribution are

$$\mu_{x_j} = \frac{\dfrac{\mu^{\pi}_{x_j}}{\sigma^{\pi}_{x_j}} + \displaystyle\sum_{i} \frac{\mu_{y_i - x_j}}{\sigma_{y_i - x_j}}}{\dfrac{1}{\sigma_{x_j}} + \displaystyle\sum_{i} \frac{1}{\sigma_{y_i - x_j}}}, \tag{4.65}$$

$$\sigma_{x_j} = \left( \frac{1}{\sigma_{x_j}} + \sum_{i} \frac{1}{\sigma_{y_i - x_j}} \right)^{-1}. \tag{4.66}$$

The goal here is to analyze how the means and variances evolve, as the $y$-to-$x$ message updates in (4.61)–(4.62) and $x$-to-$y$ message updates in (4.63)–(4.64) are computed iteratively.

Next, we provide results from [40] which say that the convergence of the variances is always guaranteed, whereas the convergence of the means is closely related to the spectral radius of an *iteration matrix* $\mathbf{\Omega}$. First, consider the convergence of the variances. Equations (4.62) and (4.64) show that the evolution of the variances is independent of the means or the observations themselves.

To simplify notation define:

$$\sigma_{i,j} = \sigma_{y_i - x_j}.$$

Let the graph be for a system with $n$ input variables $\{x_1, \ldots, x_n\}$ and also $n$ output variables $\{y_1, \ldots, y_n\}$. For iteration $t$ define the vector of variances of messages from $y$ to $x$ nodes

$$\mathbf{v}^{(t)} = \begin{bmatrix} \sigma_{1,1} & \ldots & \sigma_{1,n} & \sigma_{2,1} & \ldots & \sigma_{2,n} & \ldots & \sigma_{n,1} & \ldots & \sigma_{n,n} \end{bmatrix}^T.$$

The following result is proven in [40], and is described in detail (but without proof) in [41]. This result, and the next one, make the simplifying assumption that the variances of all the observation variables (the $y_i$s) are unity.

**Theorem 4.1** *The sequence of vectors $\mathbf{v}^{(t)}$ always converges to a unique fixed point for any $\mathbf{v}^{(0)}$, i.e.,*

$$\lim_{t \to \infty} \mathbf{v}^{(t)} = \mathbf{v}^*.$$

*Moreover, the sequence is monotonically decreasing under the initialization* $\sigma^{(0)}_{x_j - y_i} = 1$.

Next, consider the convergence of the means in the updates in (4.61) and (4.63) assuming that the variance messages are fixed to the converged values:

$$\mathbf{v}^{(t)} = \mathbf{v}^*.$$

Define

$$\mu_{i,j} = \mu_{y_i - x_j} \tag{4.67}$$

and for iteration $t$

$$\mathbf{m}^{(t)} = \begin{bmatrix} \mu_{1,1} & \dots & \mu_{1,n} & \dots \mu_{n,1} & \dots & \mu_{n,n} \end{bmatrix}^T$$

the channel matrix

$$\tilde{\mathbf{H}} = \text{diag}\{\tilde{h}_{1,1}, \dots, \tilde{h}_{1,n}, \dots, \tilde{h}_{n,1} \dots, \tilde{h}_{n,n}\},$$

where $\tilde{h}_{i,j}$ is the channel coefficient from source $x_j$ to observation $y_i$. Let $\mathbf{\Omega}$ be

$$\mathbf{\Omega} = \tilde{\mathbf{H}}^{-1}(\mathbf{\Sigma}_x - \mathbf{I})\tilde{\mathbf{H}}(\mathbf{I} + \mathcal{D}(\mathbf{\Sigma}_f \mathbf{V}^{*-1}\mathbf{\Sigma}_f) - \mathbf{V}^{*-1})^{-1}(\mathbf{\Sigma}_f - \mathbf{I})\mathbf{V}^{*-1}, \tag{4.68}$$

where

$$\begin{aligned} \mathbf{\Sigma}_x &= \text{diag}\{\mathbf{1}_{n\times n}, \dots, \mathbf{1}_{n\times n}\}, \\ \mathbf{V}^* &= \text{diag}\{\mathbf{v}^*\}, \\ \mathbf{\Sigma}_f &= \begin{bmatrix} \mathbf{I}_n & \mathbf{I}_n & \dots & \mathbf{I}_n \\ \vdots & \vdots & & \vdots \\ \mathbf{I}_n & \mathbf{I}_n & \dots & \mathbf{I}_n. \end{bmatrix}, \end{aligned}$$

$\mathbf{1}_{n\times n}$ is an $n \times n$ block of ones, $\mathbf{I}_n$ is the $n \times n$ identity matrix, and $\mathcal{D}(\cdot)$ is the operator defined as $\mathcal{D}(\mathbf{A}) = \text{diag}\{A_{11}, A_{22}, \dots, A_{nn}\}$ for a $n \times n$ matrix $\mathbf{A}$.

The following theorem provides a necessary and sufficient condition for the convergence of the means. It was proven in [40] where it was shown to follow from Theorem 5.3 in [7].

**Theorem 4.2** *The sequence of vectors* $\mathbf{m}^{(t)}$ *converges to the fixed point*

$$\mathbf{m}^* = (\mathbf{I} + \mathbf{\Omega})^{-1}\tilde{\mathbf{H}}^{-1}\mathbf{y}$$

*for any* $\mathbf{m}^{(0)}$ *if and only if the spectral radius* $\rho(\mathbf{\Omega}) < 1$.

This theorem confirms that in some scenarios, where the spectral radius condition is not met, the sum–product algorithm will not converge. When the condition is violated it is necessary to switch to another form of preprocessing, perhaps at a lower data rate using SCP – or one can declare an outage. Note that the

same issue arises for power control algorithms, where instability can also occur if the system loading is too high. Preliminary investigation of practical methods to deal with this issue have been examined in [40], but much further work, both practical and theoretical, is required to characterize properly the conditions of convergence in more general settings, and to determine what to do about lack of convergence when it arises.

We have undertaken extensive numerical experiments using different channel parameter values. For a deterministic channel model, where the channel coefficient for a mobile user is 1 to its own BS and $\alpha$ (cross-coupling factor) to an adjacent cell's BS (symmetric deterministic channel model), we have observed the following for the case of Gaussian symbols: at realistic SNRs less than 30 dB, and cross-coupling factors less than 1/2 (the typical cross-coupling between two adjacent cells is quite low, usually less than 1/2), we have never observed lack of convergence of the sum–product algorithm. However, convergence is not guaranteed at higher levels of the cross-coupling factor, when the SNR is high. Similar convergence problems were also observed for the discrete symbol model, when the deterministic channel described above was utilized.

On the other hand, the convergence problems are greatly mitigated when we replace the deterministic channel gains with independent, Rayleigh fading gains with the same means as above. Thus, even if the network is symmetric with respect to average gains, convergence problems almost completely disappear if the instantaneous gains are random (e.g., Rayleigh distributed).

From our numerical experiments, we found that when the sum–product algorithm failed to converge, it was typically due to a symmetric network realization of channel coefficients. However, symmetry can be perturbed by noise. If the SNR is sufficiently low, the realizations of noise amplitudes are large enough to perturb the symmetry of the network. On the other hand, when the channel coefficients are modeled as independent random variables (e.g., complex Gaussian), the system lacks symmetry with high probability. In other words, the probability of a realization of $\mathbf{h}$ that is symmetric enough to cause failure of convergence is extremely small.

In a practical system, the channels of different mobile users are indeed independent, due to the fact that they are located in different cells. The simulations in Figures 4.11–4.13 are for such a scenario: in each simulation realization, independent samples of channel coefficients $h_{i,j}(m,n)$ are generated. In those figures, no error floors are observed, even for very low uncoded bit error rates. There may still be an error floor, but it is too low to be detected. Note that if there were a set of channel realizations that always cause convergence failure, irrespective of SNR, then the probability of such a set must be very small, as it would provide an error floor at this probability. We emphasize that here convergence means the convergence to the true posterior probabilities of the symbols, not to the true values of the symbols. The true APP can still result in an incorrect estimate of the symbol, however if the true APP can be attained, the lowest possible error rate is obtained.

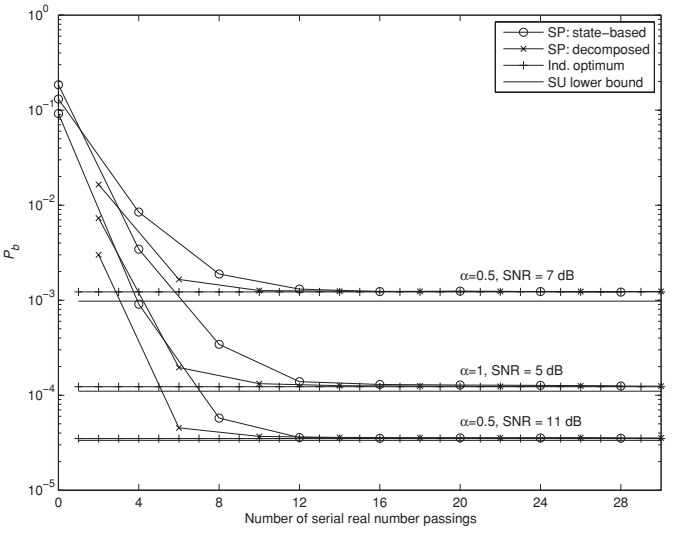


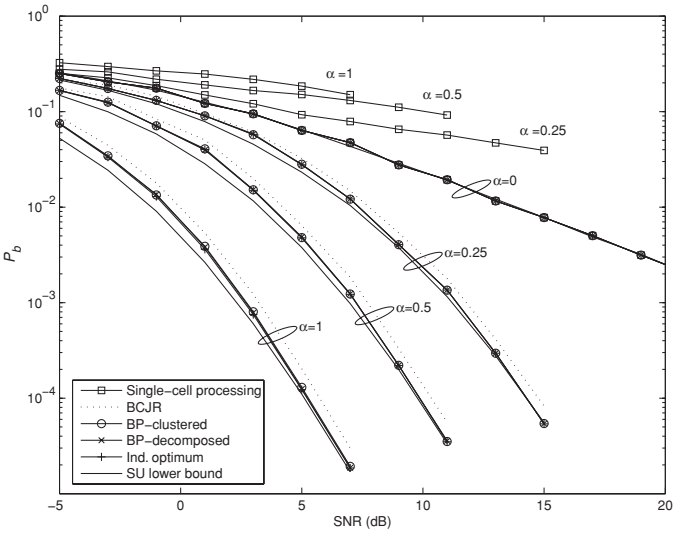

Figure 4.11. Probability of error of the algorithms and the single-user lower bound as a function of the number of serial real number transmissions between BSs for node $(2, 2)$ of a $4 \times 4$ network (© 2008 IEEE).

In the next section, we present some numerical results that indicate the relative performance of the different schemes we have proposed so far.

### 4.4.6 Numerical results

We now present some numerical results [5] comparing the aforementioned approaches for rectangular cellular arrays given in Figures 4.11, 4.12, and 4.13. In these simulations, a fading channel model is considered where each $h_{i,j}(m, n)$ is complex Gaussian distributed with zero mean and variance 1 if $(m, n) = (0, 0)$ and variance $\alpha^2$ otherwise. Thus $\alpha^2$ represents the average power of the intercell interference (ICI) from each neighbor. The additive noise $z_{i,j}$ is complex Gaussian with zero mean and variance $\sigma^2$. The signal to noise ratio (SNR) is defined as $1/\sigma^2$. The transmitted symbols are binary and from the set $\{-1, 1\}$. For comparison, the performance of individually optimum multiuser detection and the single-user lower bound is also given. The single-user lower bound is the performance of a system with a single user (with no interference) and multiple receiving base stations. Each message passing requires an amount of serial real number transmissions among base stations, and in Figure 4.11 the performance is shown as a function of those transmissions. Figure 4.11 illustrates that performance very close to the interference-free case can be achieved after 3–4 message passing steps, with the decomposed model outperforming the clustered approach.

In Figure 4.12, the performance of the optimum receiver for a base station that cannot communicate with other base stations is also shown. For the cooperative




3

Figure 4.12. Probability of error of the algorithms and the single-user lower bound as a function of the SNR for cell $(2, 2)$ of a $4 \times 4$ network (© 2008 IEEE).

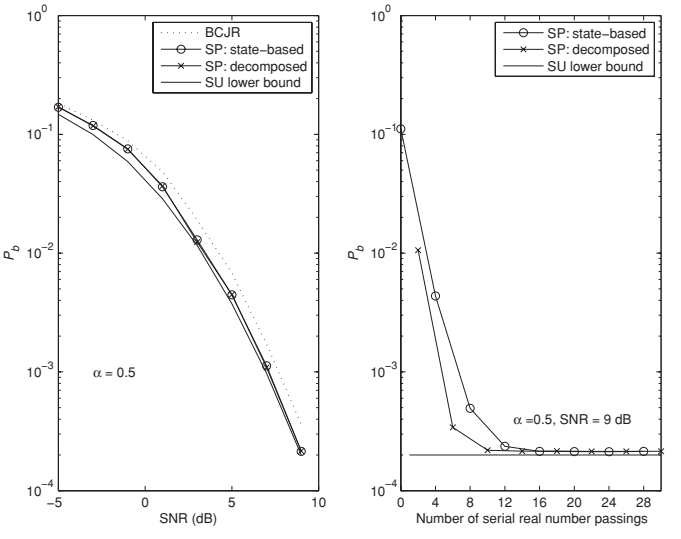



Figure 4.13. Probability of error of the algorithms and the single-user lower bound as a function of the SNR and as a function of number of real number transmissions in series for cell $(10, 10)$ of a $20 \times 20$ network (© 2008 IEEE).

cases, the values plotted are the error probabilities after enough iterations have occurred to satisfy the convergence criterion. As the ICI power increases, it is observed that the distributed decoding algorithms not only can handle the ICI, but they can also exploit the extra energy and diversity provided by it.

The error rate performance of the sum–product algorithms on the clustered and decomposed graphs is very close to the single-user lower bound at low error probabilities. Thus, we observe a very large gain is possible from local message passing to reduce the ICI. The BCJR algorithm, implemented iteratively on the columns and rows of the rectangular array, does not perform as well as the sum–product algorithms; a 0.5 dB gap is observed.

In Figure 4.13 performance is shown for a larger network. We observe that the performance on this $20 \times 20$ network is essentially the same as in the smaller network of size $4 \times 4$. The observation that the speed of convergence remains roughly the same is explained by the fact the ICI is a local effect even though the overall network size is growing.

*Simplification of messages sent by factor nodes*

In the decomposed graph approach, there are two types of computations: computation of variable-to-factor node messages in (4.19), and the computation of factor-to-variable node messages in (4.20). Upon examining these two types of messages, one sees that the main cause of computation complexity of this approach is the computation of the latter: factor-to-variable node messages.

In [9], a simplification of the messages sent by the factor nodes was proposed. The key idea here is to recognize that the message $\mu_{y_i - x_j}(x_j)$ is the individually optimal soft MUD of $x_j$ given the observation $y_i$ and the prior distributions of $n_{y_i} \backslash \{x_j\}$. If this is computationally unacceptable, suboptimal MUD methods can be substituted for this purpose. An arbitrary choice of MUD may, however, not be suitable. The MUD should be able to incorporate the prior distributions of $n_{y_i} \backslash \{x_j\}$ to produce a posterior distribution of $x_j$. If the MUD uses prior distributions of all $n_{y_i}$, then the prior information $x_j$ should be canceled in the message to $x_j$. This is because the information received from a node is not fed back to that node in factor graph methods. In [9], an iterative groupwise MUD was considered.

### 4.4.7 Ad-hoc methods utilizing turbo principle

One approach to BS cooperation is to have the BSs share their soft (or hard) bit estimates and reconstruct the interference components at the output. After subtracting the interference components from the observation, the BSs repeat the decoding. This approach was suggested in [37] for a two-cell model. In this model, the received signal at a BS comprises a desired signal component and an interference component from other-cell mobiles. For a convolutionally coded system, the BSs perform single-user decoding, ignoring the interference component completely. Then the soft bit estimates are shared between the BSs, which use this information to reconstruct and subtract the interference components from their received signals. Repeating this procedure, an iterative "turbo" BS

cooperation method is obtained. The performance of this method was investigated for various quantization strategies and constellation sizes in [37]. Note, however, that this iterative approach is ad hoc, and is not based on graph algorithms. The turbo principle and sharing soft information with the adjacent BSs was also considered in [28]. Again, coding is included in the interference reconstruction and cancellation in [28]. The turbo principle is utilized for a general cellular network. The BS cooperation methods in [28, 37] are not explicitly based on algorithms on graphs. They are ad-hoc implementations of the turbo principle among BSs which exchange soft decisions in order to improve their decisions.

### 4.4.8 Hexagonal model

All of the methods described for the rectangular cellular array model can be extended to the more realistic hexagonal array model. The positioning of the cells is shown in Figure 4.14: for example, the decomposed factor graph for this model is as in Figure 4.15.

## 4.5 Distributed transmission in the downlink

So far our focus has been on BS cooperation for the uplink of a wireless communication network. In this section, we will shift our attention to the downlink of a wireless network. The scenario where a BS simultaneously transmits independent information to multiple uncoordinated users over the wireless channel can be classified as a broadcast channel (BC) . We will first summarize the main information-theoretic results for a MIMO (vector) BC and present some practical transmission techniques proposed in the literature. Since wireless communication networks for commercial applications are multicellular, we will then briefly discuss the impact of ICI and some possible solutions. Finally, we will present turbo BS cooperation in the downlink as a practical approach for providing high spectral efficiency in the presence of ICI.

### 4.5.1 Main results for the downlink of a single-cell network

Consider the downlink of a single-cell MIMO network where a BS with $n$ antennas is transmitting information to $m$ users each with a single antenna. Assuming the channel is flat fading with no mobility, the vector of received signals at the users is modeled as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{z}, \tag{4.69}$$

where $\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_m \end{bmatrix}^T$ denotes the vector of received signals at the users, $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$ denotes the transmitted signal vector, and $\mathbf{H}$

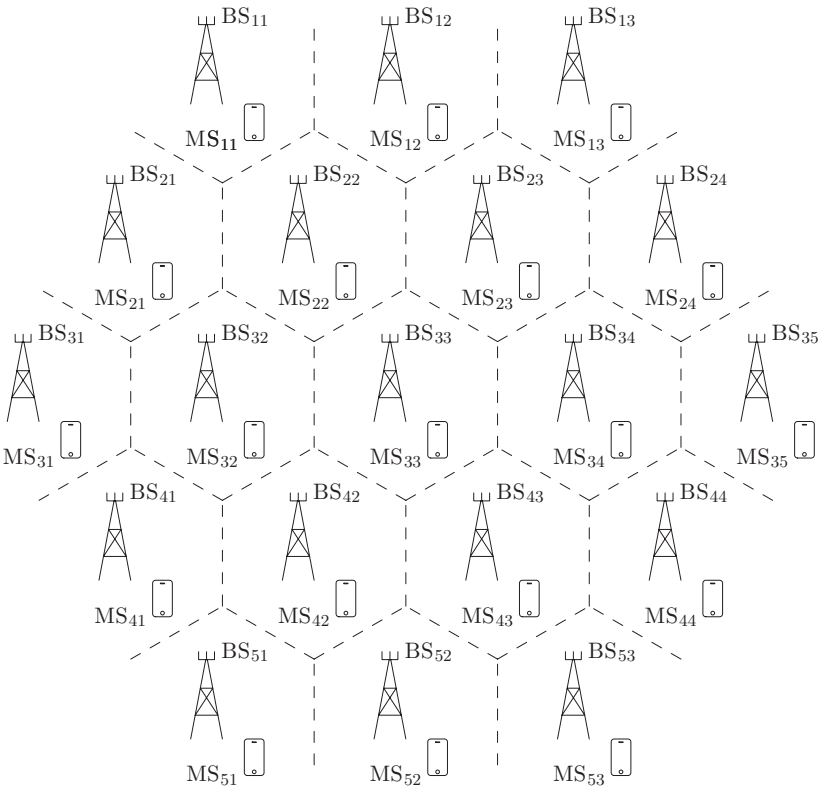



Figure 4.14. Hexagonal cellular array model. The cells are positioned on a hexagonal grid. Each cell has one active MS. The signal transmitted in one cell is received at that cell, and also six neighboring cells (except for edge cells). Dashed lines show boundaries between cells.

from the $j$th antenna of the BS to the $i$th user. Let $\mathbf{h}_i$ denote the $1 \times n$ channel vector for user $i$. Then the $(i, j)$ entry of $\mathbf{H}$ can be written $h_i(j)$. The elements of the noise vector, $\mathbf{z}$, are assumed to be independent zero-mean Gaussian distributed random variables. Unless stated otherwise, we assume that both the base stations and the users have perfect CSI.

This system model is classified as a BC in network information theory [14]. Even though the capacity region of a general BC is still an open problem, for the special case of a degraded BC superposition coding [10] is shown to achieve the capacity region. However, when the transmitter has more than one antenna, i.e., $n > 1$, the system can in general no longer be modeled as a degraded BC. Rather, the capacity region of this vector Gaussian BC is shown to be equal to the dirty paper coding (DPC) [12] rate region [56]. DPC is a nonlinear coding technique based on the observation that if the Gaussian interference is known noncausally at the transmitter but not the receiver, under a transmit power constraint, the effect of the interference can be precanceled. In the case of a vector Gaussian BC

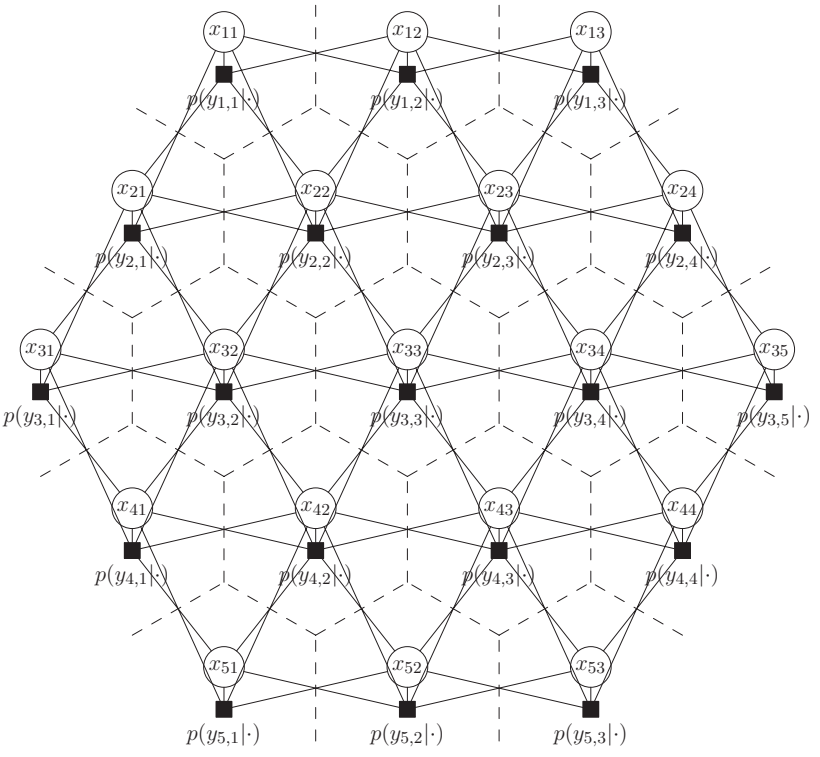



Figure 4.15. Factor graph for the decomposed probabilistic model for the hexagonal cellular array model. Dashed lines show boundaries between cells. The computations of the nodes within a cell are done by the BS of that cell. Any message passing through a cell boundary corresponds to actual message passing between corresponding BSs.

under perfect CSI the BS can precalculate noncausally the interference created by one user to the other. In this case for an arbitrary ordering of the users, using the DPC technique it is possible to encode the information of a user such that it is not affected by the interference caused by previously encoded users.

The capacity region of a vector Gaussian BC described in (4.69) under transmit covariance constraint $\mathbf{Q} = E[\mathbf{x}\mathbf{x}^H] \preceq \mathbf{S}$ for positive semidefinite $\mathbf{S}$ (where $\mathbf{A} \preceq \mathbf{B}$ implies that $\mathbf{B} - \mathbf{A}$ is a positive semidefinite matrix) is given as [56]

$$\mathcal{C} = \text{qhull}\left\{\bigcup_{\pi\in\Pi} \mathcal{R}(\pi, \mathbf{S}, \sigma^2, \mathbf{H})\right\}, \tag{4.70}$$

where qhull denotes the convex closure of the sets, $\pi$ is an arbitrary permutation of user indices, $\Pi$ corresponds to the set of all possible user permutations, $\sigma^2$

denotes the variance of the elements of the noise vector $\mathbf{z}$, and

$$\mathcal{R}(\pi, \mathbf{S}, \sigma^2, \mathbf{H}) = \left\{ (R_1, \ldots, R_m) \middle| R_i = \log \left( \frac{\mathbf{h}_i (\sum_{l=1}^{i} \mathbf{B}_{\pi(l)}) \mathbf{h}_i^H + \sigma^2}{\mathbf{h}_i (\sum_{l=1}^{i-1} \mathbf{B}_{\pi(l)}) \mathbf{h}_i^H + \sigma^2} \right) \right.$$
$$\left. \text{for some } \{\mathbf{B}_1, \ldots, \mathbf{B}_m\} \text{ such that } \mathbf{B}_i \succeq 0 \; \forall i \text{ and } \mathbf{S} \succeq \sum_{i=1}^{m} \mathbf{B}_i \right\}, \tag{4.71}$$

where $\mathbf{h}_i$ is the $1 \times n$ channel vector for user $i$ and $\pi(l)$ denotes the index of the user encoded in the $l$th position.

Correspondingly, the sum capacity of the vector Gaussian BC under total transmit power constraint $P$ is computed as

$$C_{\text{sum}} = \max_{\mathbf{Q}: \mathbf{Q} \succeq 0, \text{tr}(\mathbf{Q}) \leq P} \log(|\sigma^2 \mathbf{I}_m + \mathbf{H}\mathbf{Q}\mathbf{H}^H|), \tag{4.72}$$

where $\mathbf{I}_m$ denotes an identity matrix of size $m$, $|\cdot|$ denotes a matrix determinant and $\text{tr}(\cdot)$ denotes the trace operator. In [24] it was shown that CSI at the transmitter is essential in achieving the capacity gains of a vector Gaussian BC and at high SNR, with perfect CSI at the BS, the sum capacity scales linearly with $n$ provided that $m > n$.

The capacity achieving transmission strategy is a combination of nonlinear DPC and linear precoding (beamforming). The difficulties in implementing a practical DPC encoder have led to most attention being focused on suboptimal linear precoding schemes. In linear precoding the user data symbols are mapped to the transmitted signal vector via

$$\mathbf{x} = \mathbf{T}\mathbf{d}, \tag{4.73}$$

where $\mathbf{d}$ is the $m \times 1$ vector of data symbols with $d_j$, the data symbol intended for user $j$, taken from a finite constellation with $E[|d_j|^2] = 1$ and where the $i$th column of the $n \times m$ beamforming matrix, $\mathbf{t}_i$, is the beamforming vector for user $i$. It should be noted that the power allocated to the $i$th user is given by $P_i = \mathbf{t}_i^H \mathbf{t}_i$.

One approach for dealing with multiuser interference is to use zero-forcing (ZF) beamforming, where $\mathbf{T}$ satisfies $\mathbf{H}\mathbf{T} = \mathbf{D}$ with $\mathbf{D}$ being a diagonal matrix. One solution is

$$\mathbf{T} = \mathbf{H}^H (\mathbf{H}\mathbf{H}^H)^{-1} \mathbf{D}, \tag{4.74}$$

where $\mathbf{D}$ is determined according to the power constraint. The drawback of ZF beamforming is that transmit power might not be used as efficiently when attempting to cancel multiuser interference completely.

An alternative linear precoding scheme is regularized channel inversion (RCI) where the beamforming matrix is of the form

$$\mathbf{T} = \mathbf{H}^H (\mathbf{H}\mathbf{H}^H + \beta \mathbf{I})^{-1} \mathbf{D}, \tag{4.75}$$

where $\mathbf{D}$ is determined according to the power constraint and $\beta$ is a regularization parameter. For small $\beta$ the RCI beamformer approaches the ZF beamformer while for large $\beta$ the RCI beamformer tends towards the maximal ratio combining beamformer, the beamformer that maximizes the received signal strength while ignoring the resultant interference.

The power allocation to users is done based on an optimization criterion under a power constraint. Optimization problems considered in the literature typically include maximization of the sum rate under a total power constraint and minimization of the total transmit power under individual minimum rate requirements for each user. However, a total transmit power constraint might result in an unbalanced power output over the transmit antennas, which is undesirable in practical systems since each antenna is limited by the linear region of the power amplifiers in its own RF chain. Power allocation based on more practical per antenna power constraints has been considered [60].

Optimization problems involving RCI and related beamformers are challenging since the beamformer vectors of the users are coupled. An alternative design criterion is to maximize the signal-to-leakage-and-noise ratio (SLNR), where leakage refers to the interference caused by the user considered to other users [46]. Leakage-based precoding is attractive since the optimization problem becomes decoupled.

There are several works proposing practical nonlinear precoding techniques based on the DPC idea [16, 25, 31]. These works mainly use vector quantization or trellis/lattice precoding approaches to achieve a performance close to DPC.

### 4.5.2 Main results for downlink of a multicellular network

In a multicellular network where several BSs simultaneously serve the users in their respective cells, cochannel interference is the major performance limiting factor. This is especially true for users near cell boundaries. This communication scenario is referred to as the interference channel [2]. The capacity region of a general interference channel is still an open problem. There are some results for the extreme cases of strong interference [13] and weak interference [6]. There are also several inner and outer bounds, the best-known inner bound for discrete memoryless interference channels being the Han–Kobayashi bound [22].

As discussed in Section 4.1, the traditional method for mitigating cochannel interference in multicellular networks has been to use frequency planning such that neighboring cells use different frequency bands for transmission. If a sufficiently low frequency-reuse factor is used, one can ignore the effect of cochannel interference and the communication scenario is reduced to a number of independent single-cell downlink problems. However, as the data rate requirements for next generation wireless networks increase, it has become evident that new paradigms to mitigate cochannel interference are required.

In order to increase the spectral efficiency of the system a frequency-reuse factor of 1 can be used, i.e., the whole frequency band is utilized simultaneously

in every cell. However, this results in a reduction in the achievable data rates due to interference, especially for users near cell edges. At this point BS cooperation in the downlink is an attractive solution which actually takes advantage of the interference to increase the spectral efficiency of the system.

Assuming that the BSs are connected to each other with high capacity links over the backhaul, the cooperative system can be viewed as a virtual MIMO BC with macrodiversity, i.e., transmit antennas are distributed geographically. In that case the problem is reduced to downlink transmission in a single-cell network as discussed in Section 4.5.1, the only difference being that the transmission schemes have to be implemented in a distributed manner.

Several works have considered simple and suboptimal linear precoding schemes for the cooperative downlink. In [18], performance gains of cooperative zero-forcing beamforming were analyzed. In [26] several linear precoding techniques for downlink BS cooperation were compared in terms of sum rate per cell in the asymptotic regime. Most of the literature on downlink BS cooperation assumes that the backhaul is very high capacity. In [36], however, a finite capacity backhaul was considered and the performances of several transmission schemes involving DPC compared. Another assumption made in most of the literature is that the BSs can perfectly synchronize their transmissions. In practice, network-wide synchronization is very difficult to achieve. In [62] it was demonstrated that even though BSs can perform timing advancement perfectly such that the transmissions from different BSs arrive at the user at the same time, the interference is inevitably asynchronous. This asynchronicity causes significant performance degradation for linear precoding techniques. The techniques considered are then modified to better handle asynchronous interference.

### 4.5.3 BS cooperation schemes with message passing

As we have highlighted previously, cooperative schemes that can be implemented in a distributed manner offer several advantages over schemes requiring a central processing unit. Firstly, distributed schemes are more robust since they do not have a central processing unit as a single point of failure. Secondly, they are more scalable since schemes requiring a central processing unit require new BSs to be connected to the central processing unit as the network expands. Therefore it is of great interest to develop cooperative transmission and resource allocation schemes that only require local information exchange between BSs.

In this section, we discuss several iterative BS cooperation algorithms that require information exchange between BSs. We first focus on distributed beamforming and power allocation schemes in the literature. We then present a graph-based approach that requires only local information exchange between BSs.

*Distributed beamforming and power allocation schemes*

In [15] an iterative algorithm for computing the optimal beamforming vectors and power allocation was presented for a cooperative multicellular system. The

algorithm iteratively finds the solution of the following optimization problem:

$$\text{minimize} \sum_{j=1}^{b} \alpha_j \text{tr}(\mathbf{T}_j^H \mathbf{T}_j), \tag{4.76}$$

$$\text{subject to: } \Gamma_i \geq \gamma_j, \quad i = 1, 2, \ldots, m, \tag{4.77}$$

where $\mathbf{T}_j$ is the $n \times m_j$ beamforming matrix used at the BS in the $j$th cell with $m_j$ users, $b$ is the number of BSs (cells) in the network each with $n$ transmit antennas, $\alpha_j$ is the weighting factor of the transmit power of the $j$th BS, $\gamma_j$ denotes the target SINR constraint for user $i$, and $\Gamma_i$ is the SINR of user $i$ expressed as

$$\Gamma_i = \frac{|\mathbf{h}_{i,j(i)} \mathbf{t}_{j(i),i}^H|^2}{\sum\limits_{\substack{k=1 \\ k \neq i}}^{m} |\mathbf{h}_{i,j(k)} \mathbf{t}_{j(k),k}^H|^2 + \sigma^2}. \tag{4.78}$$

In (4.78) $\mathbf{h}_{i,j}$ denotes the $1 \times n$ channel vector between user $i$ and BS $j$, $j(i)$ denotes the index of the cell at which user $i$ is located, $\mathbf{t}_{j(i),i}^H$ is the beamforming vector used for user $i$ by the BS in its cell, and $\sigma^2$ is the variance of the Gaussian noise at the user $i$.

Note that in this scheme BSs do not jointly transmit data to all the users in the network, rather each BS transmits data to the users in its cell. However, the beamforming vectors used by BSs are jointly chosen to achieve the target SINRs of the users, taking the ICI into account while minimizing the weighted transmit power. The iterative algorithm utilizes uplink–downlink duality by solving the Lagrangian dual of the optimization problem.

Alternatively, in [11] an iterative joint transmission and power allocation scheme was proposed. In the joint transmission scheme considered, a multicell precoding approach is employed where the data of each user are available at all BSs and all BSs simultaneously serve each user using a beamforming vector. In this case the transmitted signal vector at BS $j$ is expressed as $\mathbf{x}_j = \mathbf{T}_j \mathbf{d}$, where $\mathbf{d}$ is the data vector of all the users in the network and the $i$th column of beamforming matrix $\mathbf{T}_j$ is the beamforming vector used by BS $j$ for user $i$.

The optimization problem of maximizing the sum rate of the users under individual transmit power constraints at each BS is difficult to implement in a distributed manner and requires channel knowledge to be shared by all BSs. As a result, the authors proposed a heuristic beamforming and power allocation scheme that is fully distributed and requires only statistical channel knowledge. The scheme utilizes the leakage-based beamforming approach described in Section 4.5.1 to simplify the optimization of the beamforming vectors. Furthermore, the beamforming vectors were selected to maximize the average SLNR averaged over channel realizations. The power allocation proposed is also heuristic, allocating more of the transmit power to the user with the best channel gain.

In [61], an iterative beamforming scheme for multicellular networks is proposed. The system model assumed is similar to the one in [15], where BSs do not employ multicell precoding, each BS serves a single user in its cell using a beamforming vector and the beamforming vectors are chosen such that observed ICI is taken into account. The proposed scheme can be implemented in a distributed manner as it only requires local channel knowledge. The idea is to select the beamforming vector for each BS as a linear combination of the zero-forcing beamformer and maximum ratio combining beamformer (which maximizes the received signal power ignoring the interference generated at other users). The combining weights are iteratively updated based on feedback from the users, until a Pareto optimum point is reached.

*Cooperative beamforming based on factor graphs*

We now present a cooperative downlink beamforming approach that can be implemented in a truly distributed manner based on message passing on graphs. We will first demonstrate how the downlink beamforming problem can be formulated as a virtual LMMSE estimation problem [42]. Once this is done we will be able to make use of the message passing algorithms developed for the uplink in the earlier parts of this chapter.

We focus on a cooperative downlink beamforming scenario where a cellular network with $n$ cells employs an orthogonal multiple access scheme within each cell so that users do not experience intracell interference. However, since a frequency reuse factor of 1 is used, users will experience interference from neighboring cells. For simplicity, it is assumed that each BS and user has only one antenna. The vector of received signals at the users is as given in (4.69) where the data vector of the users is mapped to the transmitted signal vector using joint linear precoding as in (4.73).

We wish to implement regularized channel inversion beamforming as discussed in Section 4.5.1. Define

$$\hat{\mathbf{T}} = \mathbf{H}^H (\mathbf{H}\mathbf{H}^H + \beta \mathbf{I})^{-1}, \tag{4.79}$$

where $\beta$ is a regularization parameter [45], and the beamforming matrix is written as

$$\mathbf{T} = \hat{\mathbf{T}}\mathbf{D}. \tag{4.80}$$

The diagonal matrix $\mathbf{D}$ denotes the power allocation matrix whose diagonals correspond to power allocated to the corresponding user under a given power (total or per BS) constraint.

The key observation is that defining $\hat{\mathbf{d}} = \mathbf{D}\mathbf{d}$, the transmitted signal $\mathbf{x} = \hat{\mathbf{T}}\hat{\mathbf{d}}$ can be seen as the LMMSE estimate of some vector $\mathbf{u}$ under the signal model

$$\hat{\mathbf{d}} = \mathbf{H}\mathbf{u} + \mathbf{w}, \tag{4.81}$$

where $\mathbf{u}$ and $\mathbf{w}$ are $n \times 1$ vectors of i.i.d. random variables with zero mean and variance 1 and $\beta$, respectively. Note that $\mathbf{u}$ and $\mathbf{w}$ have no physical relationship

with the original beamforming problem and also that the problem in (4.81) is quite different to the uplink–downlink duality concept used to simplify the downlink problem [55]. With the observation that the downlink beamforming problem can be seen as a virtual LMMSE estimation problem, one can employ distributed LMMSE estimation techniques such as the distributed Kalman smoothing algorithm described in Section 4.3.2 for one-dimensional cellular networks and techniques based on the sum–product algorithms for two-dimensional cellular networks presented in Section 4.4.5.

We first consider the one-dimensional linear cellular array depicted in Figure 4.1 which models the communication scenario with BSs placed evenly on the side of a highway or subway tunnel or access points along a corridor. The channel matrix entries $h_{i,j}$ are of the form

$$h_{i,j} = \begin{cases} h_i(k), & j = i + k, \\ 0, & |i - j| > 1, \end{cases} \tag{4.82}$$

where $i, j \in \{1, 2, \ldots, n\}$ and $k \in \{-1, 0, +1\}$. One can take advantage of the local connectivity of the network to implement a distributed beamforming algorithm. Assuming that appropriate power allocation is already performed, the data symbol vector $\hat{\mathbf{d}}$ is treated as the observation vector from which the transmitted signal vector $\mathbf{x}$ is obtained as the LMMSE estimate of $\mathbf{u}$.

Due to the Markov structure of the problem, one can apply the message passing algorithm described in Section 4.3.2 for the uplink problem with Gaussian input symbols. Defining the state vector for cell $i$ as $\mathbf{s}_i = \begin{bmatrix} u_{i-1} & u_i & u_{i+1} \end{bmatrix}^T$, the state-space model is

$$\mathbf{s}_{i+1} = \mathbf{A}^f \mathbf{s}_i + \mathbf{b}^f u_{i+2}, \tag{4.83}$$

$$\mathbf{s}_{i-1} = \mathbf{A}^b \mathbf{s}_i + \mathbf{b}^b u_{i-2}, \tag{4.84}$$

$$\hat{\mathbf{d}}_i = \hat{\mathbf{h}}_i \mathbf{s}_i + w_i, \tag{4.85}$$

where $\mathbf{A}^f$, $\mathbf{A}^b$, $\mathbf{b}^f$, and $\mathbf{b}^b$ are defined in Section 4.3.2, $\hat{\mathbf{h}}_i = \begin{bmatrix} h_i(-1) & h_i(0) & h_i(1) \end{bmatrix}$ with $u_0 = u_{n+1} = h_1(-1) = h_n(+1) = 0$. Running a forward and backward Kalman estimator based on this state-space model and combining the two estimates, we obtain a forward–backward beamforming algorithm for the cooperative downlink.

Assuming the virtual data vector $\mathbf{u}$ is Gaussian distributed (an assumption we are free to make since $\mathbf{u}$ does not have a physical meaning), the LMMSE estimate is $\hat{\mathbf{u}} = E[\mathbf{u}|\hat{\mathbf{d}}]$, i.e., the conditional mean vector of the jointly Gaussian conditional density $f(\mathbf{u}|\hat{\mathbf{d}})$. As a result, BS $i$ can find the signal that it should transmit by computing the mean of the marginal distribution, $f(u_i|\hat{\mathbf{d}})$, using the sum–product algorithm described in Section 4.3.2 running on the factor graph depicted in Figure 4.6. The steps of the algorithm are summarized below [43].

*Forward-backward cooperative downlink beamforming algorithm*

- *Initialization:* The BSs at the two edges of the cellular array compute the mean vector and the covariance matrix of the distribution information, $(\hat{\mathbf{s}}_{i|i}, \mathbf{M}_{i|i})$ for $i = 1$ and $n$, as

$$\hat{\mathbf{s}}_{i|i} = \hat{\mathbf{h}}_i^H (\hat{\mathbf{h}}_i \hat{\mathbf{h}}_i^H + \beta)^{-1} \hat{d}_i \tag{4.86}$$

$$\mathbf{M}_{i|i} = \hat{\mathbf{I}}_i - \hat{\mathbf{h}}_i^H (\hat{\mathbf{h}}_i \hat{\mathbf{h}}_i^H + \beta)^{-1} \hat{\mathbf{h}}_i, \tag{4.87}$$

where

$$\hat{\mathbf{I}}_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \hat{\mathbf{I}}_n = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{4.88}$$

- *Pass the estimates:* BS $i$, $i \in \{1, \ldots, n\}$ computes the forward and the backward estimates as the required information becomes available as

$$\hat{\mathbf{s}}_{i+1|i}^f = \mathbf{A}^f \hat{\mathbf{s}}_{i|i}^f, \tag{4.89}$$

$$\mathbf{M}_{i+1|i}^f = \mathbf{A}^f \mathbf{M}_{i|i}^f {\mathbf{A}^f}^T + [i \neq n] \mathbf{b}^f {\mathbf{b}^f}^T, \tag{4.90}$$

$$\hat{\mathbf{s}}_{i-1|i}^b = \mathbf{A}^b \hat{\mathbf{s}}_{i|i}^b, \tag{4.91}$$

$$\mathbf{M}_{i-1|i}^b = \mathbf{A}^b \mathbf{M}_{i|i}^b {\mathbf{A}^b}^T + [i \neq 1] \mathbf{b}^b {\mathbf{b}^b}^T, \tag{4.92}$$

where $\hat{\mathbf{s}}_{1|1}^f = \hat{\mathbf{s}}_{1|1}$ and $\hat{\mathbf{s}}_{n|n}^b = \hat{\mathbf{s}}_{n|n}$ are described in (4.86). Then message $(\hat{\mathbf{s}}_{i+1|i}^f, \mathbf{M}_{i+1|i}^f)$ is passed to the neighboring BS on the right and $(\hat{\mathbf{s}}_{i-1|i}^b, \mathbf{M}_{i-1|i}^b)$ is passed to the neighboring BS on the left.

- *Correct the estimates:* BS $i$, $i \in \{1, \ldots, n\}$ corrects the received forward and backward estimates using the observation $\hat{d}_i$ as

$$\hat{\mathbf{s}}_{i|i}^f = \hat{\mathbf{s}}_{i+1|i}^f + \mathbf{k}_i^f (\hat{d}_i - \hat{\mathbf{h}}_i \hat{\mathbf{s}}_{i+1|i}^f), \tag{4.93}$$

$$\mathbf{M}_{i|i}^f = (\mathbf{I}_3 - \mathbf{k}_i^f \hat{\mathbf{h}}) \mathbf{M}_{i+1|i}^f \tag{4.94}$$

with

$$\mathbf{k}_i^f = \frac{\mathbf{M}_{i+1|i}^f \hat{\mathbf{h}}^H}{\beta + \hat{\mathbf{h}} \mathbf{M}_{i+1|i}^f \hat{\mathbf{h}}^H} \tag{4.95}$$

and

$$\hat{\mathbf{s}}_{i|i}^b = \hat{\mathbf{s}}_{i-1|i}^b + \mathbf{k}_i^b (\hat{d}_i - \hat{\mathbf{h}}_i \hat{\mathbf{s}}_{i-1|i}^b), \tag{4.96}$$

$$\mathbf{M}_{i|i}^b = (\mathbf{I}_3 - \mathbf{k}_i^b \hat{\mathbf{h}}) \mathbf{M}_{i-1|i}^b \tag{4.97}$$

with

$$\mathbf{k}_i^b = \frac{\mathbf{M}_{i-1|i}^b \hat{\mathbf{h}}^H}{\beta + \hat{\mathbf{h}} \mathbf{M}_{i-1|i}^b \hat{\mathbf{h}}^H}. \tag{4.98}$$

- *Combine the estimates:* Having computed both $\hat{\mathbf{s}}^f_{i|i}$ and $\hat{\mathbf{s}}^b_{i-1|i}$, they are combined using

$$\hat{\mathbf{s}}^i = \mathbf{M}_i \left( (\mathbf{M}^f_{i|i})^{-1} \hat{\mathbf{s}}^f_{i|i} + (\mathbf{M}^b_{i-1|i})^{-1} \hat{\mathbf{s}}^b_{i-1|i} \right), \tag{4.99}$$

$$\mathbf{M}_i = \left( (\mathbf{M}^f_{i|i})^{-1} + (\mathbf{M}^b_{i-1|i})^{-1} - \hat{\mathbf{I}}_i \right)^{-1}, \tag{4.100}$$

  where $\hat{\mathbf{I}}_i = \mathbf{I}_3$ for $n = 2, \ldots, n-1$ and $\hat{\mathbf{I}}_1$ and $\hat{\mathbf{I}}_n$ are defined in (4.88). The transmitted signal from the $i$th BS, $x_i$, is set to be the middle element of the vector $\hat{\mathbf{s}}_i$.

It should be pointed out that the algorithm described above has a fully distributed nature requiring only local information exchange between the BSs. Prior to running the algorithm, however, the power allocation to users is assumed to have been done ($\mathbf{D}$ is known), and this might well require network-wide knowledge of quantities such as the channel gains. Fortunately, since the channel gains typically change much more slowly than the data symbols, global sharing of the network knowledge might still be feasible at this slower time scale.

Another point we would like to emphasize is that, as stated in Section 4.3.2, since the factor graph depicted in Figure 4.1 is free of loops, the message passing algorithm described above is guaranteed to converge to the optimal solution.

It should further be noted that the delay experienced by the BSs at the edges of the cellular array grows linearly with the size of the array. In fact, the precoding delay experienced by BSs is location-dependent with the minimum delay experienced by the BS in the middle of the array. In [43] a suboptimal limited extent distributed beamforming algorithm was proposed based on the observation that due to the local connectivity structure of the channel, the information sent by BS $i$ is expected to be less important for BS $j$ if $|i-j|$ is sufficiently large. Therefore, one can achieve a fixed delay, if the extent of the information exchange between BSs is limited. In the proposed limited extent beamforming algorithm, each BS starts by computing a 'self-estimate' based on $\hat{d}_i$ using (4.86) and (4.87). Then this information is shared with both neighbors and received estimates are corrected using forward and backward correction equations (4.93)–(4.98). After $\tau$ phases of information exchange between the BSs, forward, backward and self-estimates are combined and the middle element of the state vector $\hat{\mathbf{s}}_i$ is then an approximation to $x_i$. With this algorithm, the precoding delay experienced by all BSs is $\tau$. Numerical results in [43] demonstrate the clear tradeoff between the performance and the precoding delay.

Finally, we will discuss the extension of the proposed distributed beamforming algorithm to two-dimensional networks. As an example, we will consider a hexagonal network with $n = 7$ cells, a special case of the hexagonal network depicted in Figure 4.14. As in the one-dimensional linear array case, the downlink beamforming problem can be posed as the virtual LMMSE problem of estimating a virtual data vector $\mathbf{u}$ from the observation vector $\hat{\mathbf{d}}$. Assuming the virtual data vector is Gaussian distributed, the LMMSE estimate corresponds to the mean vector

of the MAP estimate, i.e., $\mathbf{u}$ maximizing the conditional distribution $f(\mathbf{u}|\hat{\mathbf{d}})$. As a result, one can use the sum–product algorithm on the factor graph in Figure 4.15. For the downlink, the same factor graph, in which variable nodes now represent $u_i$ and the factor nodes represent the local function $f_{\hat{d}_i} = f(\hat{d}_i|\mathbf{u})$. In addition, assume that for each variable node there is a pendant factor node that is connected only to that variable node, $f_{u_i} = p(u_i) = N(u_i; \sigma_{f_{u_i}-u_i}, \mu_{f_{u_i}-u_i})$, denoting the prior distribution of the virtual data symbols.

Since all the distributions are Gaussian, the messages passed between variable and factor nodes are actually the mean vector and covariance matrix of the underlying distribution. Assuming a flooding schedule [30] where at each iteration the messages on the graph are updated simultaneously, the sum–product update equations for $k$th iteration are summarized as follows [43]:

$$\mu^{(k)}_{f_{\hat{d}_i}-u_j} = \frac{\hat{d}_i - \sum\limits_{u_l \in n_{f_{\hat{d}_i}} \setminus \{u_j\}} h_{i,l}\ \mu^{(k-1)}_{u_k - f_{\hat{d}_i}}}{h_{i,j}}, \tag{4.101}$$

$$\sigma^{(k)}_{f_{\hat{d}_i}-u_j} = \frac{\beta + \sum\limits_{u_l \in n_{f_{\hat{d}_i}} \setminus \{u_j\}} (h_{i,l})^2 \sigma^{(k-1)}_{f_{\hat{d}_i}-u_j}}{(h_{i,j})^2}, \tag{4.102}$$

$$\mu^{(k)}_{u_j - f_{\hat{d}_i}} = \sigma^{(k)}_{u_j - f_{\hat{d}_i}} \left( \sum_{f \in n_{u_j} \setminus \{f_{\hat{d}_i}\}} \frac{\mu^{(k)}_{f-u_j}}{\sigma^{(k)}_{f-u_j}} \right), \tag{4.103}$$

$$\sigma^{(k)}_{u_j - f_{\hat{d}_i}} = \left( \sum_{f \in n_{u_j} \setminus \{f_{\hat{d}_i}\}} \frac{1}{\sigma^{(k)}_{f-u_j}} \right)^{-1} \tag{4.104}$$

with initialization as

$$\mu^{(0)}_{u_j - f_{\hat{d}_i}} = 0, \qquad \sigma^{(0)}_{u_j - f_{\hat{d}_i}} = 1 \tag{4.105}$$

for all $i, j$ for which $u_j$ is connected to $f_{\hat{d}_i}$, $n_k$ denoting the set of nodes connected to node $k$ on the factor graph and $h_{i,j}$ denoting the channel gain between the $j$th BS and the user in cell $i$. After the termination conditions are met at step $m$, the transmitted signal from BS $j$ is obtained as

$$x^{(m)}_j = \sigma^{(m)}_{u_j - f_{\hat{d}}} \left( \sum_{f \in n_{u_j}} \frac{\mu^{(m)}_{f-u_j}}{\sigma^{(m)}_{f-u_j}} \right) \tag{4.106}$$

with

$$\sigma^{(m)}_{u_j - f_{\hat{d}}} = \left( \sum_{f \in n_{u_j}} \frac{1}{\sigma^{(m)}_{f-u_j}} \right)^{-1}. \tag{4.107}$$

Since the factor graph in Figure 4.15 contains loops, the convergence of the sum–product algorithm is not guaranteed. However, since the underlying distributions are all Gaussian, following the discussion in Section 4.4.5 it can be shown that the variance updates always converge [40, 41]. The necessary and sufficient condition for the convergence of the mean updates is that the spectral radius of the iteration matrix in (4.68) is strictly less than 1. As observed for the uplink problem, the convergence conditions are violated when the SNR is high and the interference created by the channel is strong. In [40], two approaches to improve the convergence of the algorithm for the downlink were proposed. In the first approach, a tunable parameter $\epsilon$ is introduced that multiplies the regularization parameter $\beta$. The value of $\epsilon$ is chosen so that the sum–product algorithm converges or converges at the desired rate. The disadvantage of this approach is that the sum–product algorithm no longer computes the desired LMMSE estimates resulting in some loss in performance. In the second approach, a switched beam system is used where a cell is divided into angular sectors and each sector is covered by a narrow beam generated by a directional antenna. Comparing the signal strength from all sectors, the BS selects the best beam to transmit data to a user. In this way interference is reduced, and the loops in the factor graph can be reduced or eliminated.

## 4.6 Current trends and practical considerations

Much work has demonstrated the huge performance gains that are possible when BSs cooperate to receive signals from mobile users on the uplink, and to send data to mobile users on the downlink. In this chapter we have examined methods for implementing BS cooperation in a distributed manner via message passing. These message passing techniques require communication between neighboring BSs only and have processing and communication requirements that remain constant with increasing network size. They are, in some sense, the natural algorithms to use in large cellular networks.

However, in terms of both theoretical analysis and practical implementation of these turbo-style approaches, we have only scratched the surface. We list below some exciting areas where much work is still to be done.

(1) *Convergence issues.* Many questions are still unanswered regarding the convergence of the sum–product algorithm on graphs with loops, as mentioned in Section 4.4.5. A more thorough treatment of convergence is required in order to understand the limitations of the distributed BS processing techniques presented in the chapter and to find methods to improve the convergence properties.
(2) *Limited backhaul traffic.* Backhaul traffic is the traffic between BSs required to implement cooperative schemes. For a practical system, the backhaul traffic is not unlimited. Given a certain limit on the backhaul traffic, can we still

get the performance gains we see in this chapter? How should the message passing techniques be modified to reduce the backhaul traffic load?

(3) *Synchronization.* Perfect synchronization of BSs in the downlink has been assumed in this chapter. We have also assumed it for the uplink, although the assumption may be less critical there. What is the impact of synchronization errors on the performance? It may be possible that a synchronization imperfection that is tolerable in conventional single-cell processing may have more degrading effects in the case of BS cooperation.

(4) *Imperfect channel information.* We have assumed that perfect CSI is available at the BS transceiver. What will be the impact of imperfections in channel estimates on the attainable gains due to base station cooperation? What will be the impact on the convergence of the proposed methods?

(5) *Coded systems.* We have not considered error control coding in this chapter. In practice, common decoders are often based on graphical methods and it would be possible to combine the factor graphs we use for joint detection of mobile users and the graphs which model the code constraints, to form one large graphical model on which message passing algorithms could be applied. What will happen to the convergence properties of the sum–product algorithm on this space-time factor graph? What are the other issues to consider for distributed decoding in a cellular system?

(6) *Distributed resource allocation.* If we are allocating channels to mobile users in order to maximize the throughput, how can this be done in a BS-cooperating network in a distributed manner?

Some excellent work has already begun to answer some of these questions: channel estimation and coding [63], limited backhaul traffic and coding [28, 37], and resource allocation [1]. However, many questions remain unanswered and numerous challenges must be overcome if we are to realize the full potential of turbo-based methods for BS cooperation.

## References

[1] A. Abrardo, P. Detti, and M. Moretti, "Message passing resource allocation for the uplink of multicarrier systems," in *Proc. of IEEE International Conference on Communications*, 2009, pp. 1–6. IEEE, 2009.

[2] R. Ahlswede, "The capacity region of a channel with two senders and two receivers," *Annals of Probability*, **2**, 1974, 805–814.

[3] E. Aktas, J. Evans, and S. Hanly, "Distributed decoding in a cellular multiple-access channel," in *Proc. of IEEE International Symposium on Information Theory*, 2004, p. 484. IEEE, 2004.

[4] E. Aktas, J. Evans, and S. Hanly, "Distributed base station processing in the uplink of cellular networks," in *Proc. of IEEE International Conference on Communications*, 2006, pp. 1641–1646. IEEE, 2006.

[5] E. Aktas, J. Evans, and S. Hanly, "Distributed decoding in a cellular multiple-access channel," *IEEE Transactions on Wireless Communications*, **7**, 2008, 241–250.

[6] V. S. Annapureddy and V. V. Veeravalli, "Gaussian interference networks: Sum capacity in the low interference regime and new outer bounds for the capacity region," *IEEE Transactions on Information Theory*, **55**, 2009, 3032–3050.

[7] O. Axelsson, *Iterative Solution Methods*. Cambridge University Press, 1994.

[8] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, **20**, 1974, 284–287.

[9] S. Bavarian and J. K. Cavers, "Reduced-complexity belief propagation for system-wide MUD in the uplink of cellular networks," *IEEE Journal on Selected Areas in Communications*, **26**, 2008, 541–549.

[10] P. P. Bergmans, "Random coding theorem for broadcast channels with degraded components," *IEEE Transactions on Information Theory*, **19**, 1973, 197–207.

[11] E. Bjornson and B. Ottersten, "On the principles of multicell precoding with centralized and distributed cooperation," in *Proc. of International Conference on Wireless Communication and Signal Processing*, 2009, pp. 6–10. IEEE, 2009.

[12] M. Costa, "Writing on dirty paper," *IEEE Transactions on Information Theory*, **29**, 1983, 439–441.

[13] M. H. M. Costa and A. E. Gamal, "The capacity region of the discrete memoryless interference channel with strong interference," *IEEE Transactions on Information Theory*, **33**, 1987, 710–711.

[14] T. Cover, "Broadcast channels," *IEEE Transactions on Information Theory*, **18**, 1972, 2–14.

[15] H. Dahrouj and W. Yu, "Coordinated beamforming for the multi-cell multi-antenna wireless system," in *Proc. of Conference on Information Sciences and Systems*, 2008, pp. 429–434. John Hopkins University Press, 2008.

[16] U. Erez and S. ten Brink, "A close to capacity dirty paper coding scheme," *IEEE Transactions on Information Theory*, **51**, 2005, 3417–3432.

[17] G. D. Forney Jr., "Codes on graphs: Normal realizations," *IEEE Transactions on Information Theory*, **47**, 2001, 520–548.

[18] G. J. Foschini, M. K. Karakayali, and R. A. Valenzuela, "Coordinating multiple antenna cellular networks to achieve enormous spectral efficiency," *IEE Proceedings – Communications*, **153**, 2006, 548–555.

[19] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*. MIT Press, 1998.

[20] R. G. Gallager, *Low-Density Parity-Check Codes*. MIT Press, 1963.

[21] A. Grant, S. Hanly, J. Evans, and R. Müller, "Distributed decoding for Wyner cellular systems," in *Proc. of Australian Communications Theory*

*Workshop*, 2004, pp. 77–81. The University of Newcastle, NSW, Australia, 2004.

[22] T. S. Han and K. Kobayashi, "A new achievable rate region for the interference channel," *IEEE Journal on Selected Areas in Communications*, **27**, 1981, 49–60.

[23] S. V. Hanly and P. Whiting, "Information-theoretic capacity of multi-receiver networks," *Telecommunication Systems*, **1**, 1993, 1–42.

[24] B. Hassibi and M. Sharif, "Fundamental limits in MIMO broadcast channels," *IEEE Journal on Selected Areas in Communications*, **25**, 2007, 1333–1344.

[25] B. M. Hochwald, C. B. Peel, and A. L. Swindlehurst, "A vector-perturbation technique for near-capacity multiantenna multiuser communication – Part II: Perturbation," *IEEE Transactions on Communications*, **53**, 2005, 537–544.

[26] S. Jin, D. N. C. Tse, J. B. Soriaga, J. Hou, J. E. Smee, and R. Padovani, "Downlink macro-diversity in cellular networks," in *Proc. of IEEE International Symposium on Information Theory*, pp. 2007, 1–5. IEEE, 2007.

[27] S. M. Kay, *Statistical Signal Processing: Estimation Theory*. Prentice Hall, 1993.

[28] S. Khattak, W. Rave, and G. Fettweis, "Distributed iterative multiuser detection through base station cooperation," *EURASIP Journal on Wireless Communications and Networking*, 2008.

[29] R. Kinderman and J. Snell, *Markov Random Fields and Their Applications*. American Mathematical Society, 1980.

[30] F. R. Kschischang, B. J. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, **47**, 2001, 498–519.

[31] S.-C. Lin and H.-J. Sun, "Practical vector dirty paper coding for MIMO Gaussian broadcast channels," *IEEE Journal on Selected Areas in Communications*, **25**, 2007, 1345–1357.

[32] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Magazine*, **21**, 2004, 28–41.

[33] H.-A. Loeliger, J. Dauwels, J. Hu, S. Korl, L. Ping, and F. R. Kschischang, "The factor graph approach to model-based signal processing," *Proceedings of the IEEE*, **95**, 2007, 1295–1322.

[34] Y. Mao and F. R. Kschischang, "On factor graphs and the Fourier transform," *IEEE Transactions on Information Theory*, **51**, 2005, 1635–1649.

[35] M. Marrow and J. K. Wolf, "Iterative detection of 2-dimensional ISI channels," in *Proc. of IEEE Information Theory Workshop*, 2003, pp. 131–134. IEEE, 2003.

[36] P. Marsch and G. Fettweis, "On base station cooperation schemes for downlink network MIMO under a constrained backhaul," in *Proc. of IEEE Global Telecommunications Conference*, 2008, pp. 1219–1224. IEEE, 2008.

[37] T. Mayer, H. Jenkac, and J. Hagenauer, "Turbo base-station cooperation for intercell interference cancellation," in *Proc. of IEEE International Conference on Communications*, 2006, pp. 4977–4982. IEEE, 2006.

[38] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as an instance of Pearl's belief propagation algorithm," *IEEE Journal on Selected Areas in Communications*, **16**, 1998, 140–152.

[39] B. L. Ng, J. S. Evans, and S. V. Hanly, "Distributed linear multiuser detection in cellular networks based on Kalman smoothing," in *Proc. of IEEE Global Communications Conference*, 2004, pp. 134–138. IEEE, 2004.

[40] B. L. Ng, Cellular networks with Cooperating base stations: Performance analysis and distributed algorithms, Ph.D. Thesis, University of Melbourne, Melbourne, Australia, 2007.

[41] B. L. Ng, J. S. Evans, and S. V. Hanly, "Distributed downlink beamforming in cellular networks," in *Proc. of IEEE International Symposium on Information Theory*, 2007, pp. 6–10. IEEE, 2007.

[42] B. L. Ng, J. S. Evans, S. V. Hanly, and D. Aktas, "Transmit beamforming with cooperative base stations," in *Proc. of IEEE International Symposium on Information Theory*, 2005, pp. 1431–1435. IEEE, 2005.

[43] B. L. Ng, J. S. Evans, S. V., Hanly, and D. Aktas, "Distributed downlink beamforming with cooperative base stations," *IEEE Transactions on Information Theory*, **54**, 2009, 5491–5499.

[44] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[45] C. B. Peel, B. M. Hochwald, and A. L. Swindlehurst, "A vector-perturbation technique for near-capacity multiantenna multiuser communication – Part I: Channel inversion and regularization," *IEEE Transactions on Communications*, **53**, 2005, 195–202.

[46] M. Sadek, A. Tarighat, and A. H. Sayed, "A leakage-based precoding scheme for downlink multi-user MIMO channels," *IEEE Transactions on Wireless Communications*, **6**, 2007, 1711–1721.

[47] C. Sankaran and A. Ephremides, "Solving a class of optimum multiuser detection problems with polynomial complexity," *IEEE Transactions on Information Theory*, **44**, 1998, 1958–1961.

[48] C. Schlegel and A. Grant, "Polynomial complexity optimal detection of certain multiple-access systems," *IEEE Transactions on Information Theory*, **46**, 2000, 2246–2248.

[49] S. Shamai and B. M. Zaidel, "Enhancing the cellular downlink capacity via co-processing at the transmitting end," in *Proc. of IEEE Vehicular Technology Conference*, 2001, pp. 1745–1749. IEEE, 2001.

[50] O. Shental, A. J. Weiss, N. Shental, and Y. Weiss, "Generalized belief propagation receiver for near-optimal detection of two-dimensional channels with memory," in *Proc. of IEEE Information Theory Workshop*, 2004, pp. 225–229. IEEE, 2004.

[51] R. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, **27**, 1981, 533–547.

[52] S. Ulukus and R. D. Yates, "Optimum multiuser detection is tractable for synchronous CDMA systems using m-sequences," *IEEE Communications Letters*, **2**, 1998, 89–91.

[53] M. Valenti and B. Woerner, "Iterative multiuser detection, macrodiversity combining, and decoding for the TDMA cellular uplink," *IEEE Journal on Selected Areas in Communications*, **19**, 2001, 1570–1583.

[54] S. Verdú, *Multiuser Detection*. Cambridge University Press, 1998.

[55] P. Vishwanath and D. N. C. Tse "Sum capacity of vector Gaussian broadcast channel and uplink-downlink duality," *IEEE Transactions on Information Theory*, **49**, 2003, 1912–1921.

[56] H. Weingarten, Y. Steinberg, and S. Shamai, "The capacity region of the Gaussian multiple-input multiple-output broadcast channel," *IEEE Transactions on Information Theory*, **52**, 2006, 3936–3964.

[57] L. Welburn, J. K. Cavers, and K. W. Sowerby, "A computational paradigm for space-time multiuser detection," *IEEE Transactions on Communications*, **52**, 2004, 1595–1604.

[58] N. Wiberg, H. Loeliger, and R. Koetter, "Codes and iterative decoding on general graphs," *European Transactions on Telecommunications*, **6**, 1995, 513–525.

[59] A. Wyner, "Shannon-theoretic approach to a Gaussian cellular multiple-access channel," *IEEE Transactions on Information Theory*, **40**, 1994, 1713–1727.

[60] W. Yu and T. Lan, "Transmitter optimization for the multi-antenna downlink with per-antenna power constraints," *IEEE Transactions on Signal Processing*, **55**, 2007, 2646–2660.

[61] R. Zakhour, Z. K. M. Ho, and D. Gesbert, "Distributed beamforming coordination in multicell MIMO channels," in *Proc. of IEEE Vehicular Technology Conference*, 2009, pp. 1–5. IEEE, 2009.

[62] H. Zhang, N. B. Mehta, A. F. Molisch, J. Zhang, and H. Dai, "Asynchronous interference mitigation in cooperative base station systems," *IEEE Transactions on Wireless Communications*, **7**, 2008, 155–165.

[63] Y. Zhu, D. Guo, and M. Honig, "Joint channel estimation and co-channel interference mitigation in wireless networks using belief propagation," in *Proc. of IEEE International Conference on Communications*, 2008, pp. 2003–2007. IEEE, 2008.