Summary of Soft-In/Soft-out (SISO) Decoding

Keith M. Chugg

November 19, 2015

1 Definitions

In the metric domain, soft information is stored in the negative-log of the probability or likelihood. Therefore, high confidence for a conditional value corresponds to a low metric. Let us consider a code with k input information bits $\{b_i\}$ and n output, coded bits, $\{c_j\}$. Index the 2^k code configurations by $m = 0, 1, \ldots 2^k - 1$. Then, we have

$$\mathrm{MI}[b_i] \stackrel{\Delta}{=} -\ln p_{b_i(u)}(b_i) \qquad b_i = 0, 1 \tag{1}$$

$$\mathrm{MI}[c_j] \stackrel{\Delta}{=} -\ln f_{z_j(u)|c_j(u)}(z_j|c_j) \qquad c_j = 0, 1 \tag{2}$$

Notice that $\mathrm{MI}[b_i = 1] > \mathrm{MI}[b_i = 0]$ means that the conditional value 0 is more likely. The input metrics on b_i measure the a-priori belief about the bit values and the input metrics on c_j correspond to channel likelihoods.

For the BPSK-AWGN channel

$$z_j(u) = \sqrt{E_c}(-1)^{c_j(u)} + w_j(u) \tag{3}$$

where $w_j(u)$ is an iid sequence of Gaussian random variables with mean zero and variance $N_0/2$. For this model, we have

$$\mathrm{MI}[c_j] = \frac{(z_j - \sqrt{E_c}(-1)^{c_j})^2}{N_0} \tag{4}$$

We are concerned only with the difference between metrics. Specifically, we can add any finite constant to the metrics for a particular variable without affecting the underlying soft-decision information. This is equivalent to multiplication by a positive constant in the probability domain. It is useful to select as a constant, the metric for the conditional value zero. Subtraction of this constant is common and we will refer to the resulting metrics as *normalized*. In normalized form, the metric of a zero conditional value is zero, so we need only specify one number for soft-decision information on a binary variable. Specifically, define the normalized metric as

$$\overline{\mathrm{MI}}[b_i] \stackrel{\Delta}{=} \mathrm{MI}[b_i] - \mathrm{MI}[b_i = 0] = -\ln\left(\frac{p_{b_i(u)}(b_i)}{p_{b_i(u)}(0)}\right) \tag{5}$$

$$\overline{\mathrm{MI}}[c_j] \stackrel{\Delta}{=} \mathrm{MI}[c_j] - \mathrm{MI}[c_j = 0] = -\ln\left(\frac{f_{z_j(u)|c_j(u)}(z_j|c_j)}{f_{z_j(u)|c_j(u)}(z_j|0)}\right) \tag{6}$$

Note that $\overline{\text{MI}}[b_i = 0] = 0$ and only $\overline{\text{MI}}[b_i = 1]$ need be stored. Also, for this reason, the notation $\overline{\text{MI}}[b_i]$ can also be used to represent this single number (i.e., $\overline{\text{MI}}[b_i = 1]$).

In normalized form, the magnitude of the metric describes the level of confidence and the sign determines the implied hard-decision. Specifically, if $\overline{\text{MI}}[b_i = 1]$ is positive, then the value $b_i = 0$ is more likely. The larger $|\overline{\text{MI}}[b_i = 1]|$ the higher the level of confidence in this decision.

Notice that for the BPSK-AWGN channel,

$$\overline{\text{MI}}[c_j = 1] = \frac{(z_j - \sqrt{E_c}(-1))^2}{N_0} - \frac{(z_j - \sqrt{E_c}(+1))^2}{N_0} = \frac{4\sqrt{E_c}}{N_0} z_j \tag{7}$$

2 The SISO Decoder

The SISO decoder is based on the assumption of an independent input sequence and a memoryless channel. The SISO processing has two stages:

1. *Combining* of input marginal metrics to compute metrics for each code configuration
2. *Marginalization* of configuration metrics to obtain output metrics on the variables associated with the code.

The combining is done by summation of the incoming metrics for the conditional value of each input/output variable associated with the given configuration. Specifically,

$$\text{M}[\text{Config} = m] = \sum_{p=0}^{k-1} \text{MI}[b_p^{(m)}] + \sum_{q=0}^{n-1} \text{MI}[c_q^{(m)}] \tag{8}$$

Note that the above holds whether or not the metrics are normalized.

The marginalization is done for each conditional value of each variable by minimizing over all configurations consistent with that conditional value. Specifically,

$$\text{MSM}[b_i = 1] = \min_{\text{Config}=m:b_i=1} \text{M}[\text{Config} = m] \tag{9}$$

$$\text{MSM}[b_i = 0] = \min_{\text{Config}=m:b_i=0} \text{M}[\text{Config} = m] \tag{10}$$

$$\text{MSM}[c_j = 1] = \min_{\text{Config}=m:c_j=1} \text{M}[\text{Config} = m] \tag{11}$$

$$\text{MSM}[c_j = 0] = \min_{\text{Config}=m:c_j=0} \text{M}[\text{Config} = m] \tag{12}$$

where MSM can be read as the minimum summed (configuration) metric. Note that we may also place this is normalized form – i.e., $\overline{\text{MSM}}[b_i] = \text{MSM}[b_i = 1] - \text{MSM}[b_i = 0]$

Finally, the convention is to place this output metric in so-called "extrinsic" form by removing the input metric from this MSM value:

$$\overline{\text{MO}}[b_i] = \overline{\text{MSM}}[b_i] - \overline{\text{MI}}[b_i] \tag{13}$$

$$\overline{\text{MO}}[c_j] = \overline{\text{MSM}}[c_j] - \overline{\text{MI}}[c_j] \tag{14}$$

Notice that the term $\overline{\text{MI}}[b_i]$ has been added into the configuration metric in (8) and then subtracted out in (13). For this reason, you will also see these operations summarized as

$$\text{MO}[b_i] = \min_{\text{Config}=m:b_i} \left\{ \sum_{p=0,p\neq i}^{k-1} \text{MI}[b_p^{(m)}] + \sum_{q=0}^{n-1} \text{MI}[c_q^{(m)}] \right\} \tag{15a}$$

$$\text{MO}[c_j] = \min_{\text{Config}=m:c_j} \left\{ \sum_{p=0}^{k-1} \text{MI}[b_p^{(m)}] + \sum_{q=0,q\neq j}^{n-1} \text{MI}[c_q^{(m)}] \right\} \tag{15b}$$

3 Generalizations

Notice that if we did not move to the metric domain, we would have obtained similar results, but with combining done via multiplication and marginalization done via maximization. For example, the probability domain equivalent of (15a) is

$$\text{PO}[b_i] = \max_{\text{Config}=m:b_i} \left\{ \prod_{p=0,p\neq i}^{k-1} \text{PI}[b_p^{(m)}] \times \prod_{q=0}^{n-1} \text{PI}[c_q^{(m)}] \right\} \tag{16a}$$

$$\text{PO}[c_j] = \max_{\text{Config}=m:c_j} \left\{ \prod_{p=0}^{k-1} \text{PI}[b_p^{(m)}] \times \prod_{q=0,q\neq j}^{n-1} \text{PI}[c_q^{(m)}] \right\} \tag{16b}$$

where $\text{PI}[\cdot] = \exp(-\text{MI}[\cdot])$ and $\text{PO}[\cdot] = \exp(-\text{MO}[\cdot])$.

It is customary to refer to the SISO processing by the combining an marginalization operators used. For example, we described in detail, the min-sum processing in Section 2. The above equation is the max-product version. The min-sum is the equivalent of the max-product processing, just carried out in the metric domain.

Recall from lecture that both the min-sum and max-product we inspired by bit-level decision rules based on the MAP sequence (codeword) decision rule. We also mentioned that one could consider minimizing the bit error probability by finding the MAP bit decision rule. This requires computation of the bit APPs, or equivalently

$$f_{b_i(u),\mathbf{z}(u)}(b_i, \mathbf{z}) = \sum_{\mathbf{b}:b_i} f(\mathbf{z}|\mathbf{b})p(\mathbf{b}) \tag{17}$$

$$= \sum_{\mathbf{b}:b_i} \left\{ \prod_{q=0}^{n-1} f(z_j|c_j(\mathbf{b})) \times \prod_{p=0}^{k-1} p(b_p(\mathbf{b})) \right\} \tag{18}$$

Notice now that the 2^k configurations are being enumerated by $\mathbf{b}$. With this observation, we see that the optimal bit decision rule corresponds to product combining of probabilities and summation

for marginalizing. This motivates the definition of the sum-product SISO rules

$$\mathrm{PO}[b_i] = \sum_{\mathrm{Config}=m:b_i} \left\{ \prod_{p=0,p\neq i}^{k-1} \mathrm{PI}[b_p^{(m)}] \times \prod_{q=0}^{n-1} \mathrm{PI}[c_q^{(m)}] \right\} \tag{19}$$

$$\mathrm{PO}[c_j] = \sum_{\mathrm{Config}=m:c_j} \left\{ \prod_{p=0}^{k-1} \mathrm{PI}[b_p^{(m)}] \times \prod_{q=0,q\neq j}^{n-1} \mathrm{PI}[c_q^{(m)}] \right\} \tag{20}$$

which differs from (16a) only by replacing the maximization operation by summation.

This sum-product processing can also be done in the metric domain. This takes the following form:

$$\mathrm{MO}[b_i] = \min_{m:b_i}{}^{*} \left\{ \sum_{p=0,p\neq i}^{k-1} \mathrm{MI}[b_p^{(m)}] + \sum_{q=0}^{n-1} \mathrm{MI}[c_q^{(m)}] \right\} \tag{21a}$$

$$\mathrm{MO}[c_j] = \min_{m:c_j}{}^{*} \left\{ \sum_{p=0}^{k-1} \mathrm{MI}[b_p^{(m)}] + \sum_{q=0,q\neq j}^{n-1} \mathrm{MI}[c_q^{(m)}] \right\} \tag{21b}$$

where the $\min^*$ operation is defined by

$$\min{}^{*}(x_1, x_2 \ldots x_n) \stackrel{\Delta}{=} -\ln(e^{-x_1} + e^{-x_2} + \cdots e^{-x_n}) \tag{22}$$

and it follows that

1. $\min^*(x, y) = \min(x, y) - \ln\left(1 + e^{-|x-y|}\right)$
2. $\min^*(x, y, z) = \min^*(\min^*(x, y), z)$

In summary, there are two basic SISO processing methods, one based on optimal codeword decision and one based on optimal bit decisions. Both can be carried out in either the natural probability domain or in the metric (negative-log-probability) domain. The SISO based on sequence optimality uses max-product processing in the probability domain and min-sum processing in the metric domain. The SISO based on bit optimality uses sum-product processing in the probability domain and $\min^*$-sum processing in the metric domain.

3.1 The Marginalization-Combining Semi-Ring

Eventually, we will be interested in algorithms for computing the SISO processing efficiently – i.e., by not computing the 2^k configuration metrics explicitly. By noting some common properties of the marginalization and combining operators, it is possible to show that many of these algorithms can easily be converted from one format to another. For example, if one has derived an efficient algorithm for computing the sum-product SISO operation, then it may be possible to convert it to the appropriate min-sum SISO by simply changing the soft information to metric form and replacing all sum operations by min operations, then replacing all products by sums. In that sense, there is only "one" algorithm that can take various forms according to the selection of the probability/metric domain and the bit/sequence optimality criterion.

In order for this transformation to hold, the algorithm must be obtained using only the semi-ring properties of the the marginalization/combining operators. These properties are described below – taken from Chugg, et.al., "Iterative Detection".

The condition underlying this duality principle is that the marginalization and combining operators considered ($\textcircled{c}$, $\textcircled{m}$), together with the ranges for the associated soft information ($\mathcal{F}$), form a *commutative semi-ring*. Specifically, ($\mathcal{F}$, $\textcircled{c}$, $\textcircled{m}$, $\mathrm{I}_c, \mathrm{I}_m$) forms a *commutative semi-ring* if

(SR1) $\textcircled{m}$ and $\textcircled{c}$ are associative and commutative on $\mathcal{F}$

(SR2) Identity elements: $\exists\ \mathrm{I}_c, \mathrm{I}_m \in \mathcal{F}$ such that $f \textcircled{c} \mathrm{I}_c = f$ and $f \textcircled{m} \mathrm{I}_m = f$ for all $f \in \mathcal{F}$

(SR3) Distributive Law: $f \textcircled{c} (g \textcircled{m} h) = (f \textcircled{c} g) \textcircled{m} (f \textcircled{c} h)$

Note that, in general, there is no inverse for the marginalization or combining operator. However, for most cases of practical interest, the combining operation is invertible. Thus, throughout this book, we assume another property, namely

- Combining Inverse: $\forall\ f \in \mathcal{F}$ and $f \neq \mathrm{I}_m$, there exists $\bar{f} \in \mathcal{F}$ such that $f \textcircled{c} \bar{f} = \mathrm{I}_c$. We denote $g \textcircled{c} \bar{f}$ by $g \textcircled{c}^{-1} f$.

We use the inverse combining operator only to simplify the presentation of some operations. Furthermore, this operator is only applied in the form $(f \textcircled{c} g \textcircled{c} h) \textcircled{c}^{-1} f$ – i.e., where it can be interpreted as operator that specifies a term be excluded from a stated combination.

The correspondence between the specific cases discussed and this general setting is summarized in Table 1.

Soft-Info(·)	$\mathcal{F}$	$\textcircled{m}$	$\textcircled{c}$	I_m	I_c	$\bar{f}$	$\textcircled{c}^{-1}$	Threshold operation
APP	$[0, \infty)$	$+$	$\times$	0	1	$1/f$	$\div$	$\arg\max$
neg-log-APP	$(-\infty, \infty]$	$\min^*$	$+$	∞	0	$-f$	$-$	$\arg\min$
Generalized-APP	$[0, \infty)$	max	$\times$	0	1	$1/f$	$\div$	$\arg\max$
MSM	$(-\infty, \infty]$	min	$+$	∞	0	$-f$	$-$	$\arg\min$

Table 1: Parameters of the semi-ring for each of the marginalization combining schemes discussed. The threshold operation is the method used to convert the given soft measure into a hard decision.