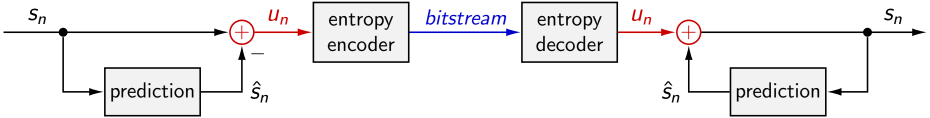
Predictive Lossless Coding



Last Lecture: Arithmetic Coding

Arithmetic Coding

- Practical realization of Shannon-Fano-Elias Coding (using standard integer arithmetic)
- No codeword table, on-the-fly encoding and decoding

Arithmetic Coding vs Huffman Coding

- For given block size N: Huffman coding is optimal, arithmetic coding is suboptimal
- Arithmetic coding is realizable for large N, while Huffman coding is not

Coding Efficiency of Arithmetic Coding

- Given probabilities and large N: Coding efficiency is very close to theoretical optimum
- In practice: Coding efficiency depends on using suitable probabilities
  - ➡ Adaptive pmf estimation during encoding and decoding
  - ➡ Using conditional pmfs (switch adaptive pmfs during encoding and decoding)

Arithmetic Coding with Adaptive Pmfs

Basic Idea

- Update pmf after encoding of each symbol
- Update pmf after decoding of each symbol

Arithmetic Coding with Adaptive Pmfs

Basic Idea

- Update pmf after encoding of each symbol
- Update pmf after decoding of each symbol

Straightforward realization

- Count occurrences N_k of alphabet letters a_k
- V-bit probabilities $p_V(a_k)$ are given by

$$p_V(a_k) = \left\lfloor 2^V \cdot \frac{N_k}{\sum_k N_k} \right\rfloor \qquad \left(\begin{array}{l} \text{rounding} \\ \text{down !} \end{array} \right)$$

- Initialization: $\forall k, \quad N_k = 1$

Arithmetic Coding with Adaptive Pmfs

Basic Idea

- Update pmf after encoding of each symbol
- Update pmf after decoding of each symbol

Straightforward realization

- Count occurrences N_k of alphabet letters a_k
- V-bit probabilities $p_V(a_k)$ are given by

$$p_V(a_k) = \left\lfloor 2^V \cdot \frac{N_k}{\sum_k N_k} \right\rfloor \qquad \left(\begin{array}{l} \text{rounding} \\ \text{down !} \end{array} \right)$$

- Initialization: $\forall k, \quad N_k = 1$

Controlling Adaptation Speed

- One possibility:
  Rescale counts after sum exceeds some limit

Arithmetic Coding with Adaptive Pmfs

Basic Idea

- Update pmf after encoding of each symbol
- Update pmf after decoding of each symbol

Straightforward realization

- Count occurrences N_k of alphabet letters a_k
- V-bit probabilities $p_V(a_k)$ are given by

$$p_V(a_k) = \left\lfloor 2^V \cdot \frac{N_k}{\sum_k N_k} \right\rfloor \qquad \left(\begin{array}{l} \text{rounding} \\ \text{down !} \end{array} \right)$$

- Initialization: $\forall k, \quad N_k = 1$

Controlling Adaptation Speed

- One possibility:
  Rescale counts after sum exceeds some limit

```
class Pmf // example for adaptive pmf implementation
{
public:
  Pmf( int Vbits, int numLetters, int maxSumCounts )
  : V          ( Vbits )
  , maxSum     ( maxSumCounts   ) // adaption speed
  , sumCounts( numLetters      )
  , counts     ( numLetters, 1 ) // all counts = 1
  {}

  int operator[] ( int index ) const  {
    return ( counts[ index ] << V ) / sumCounts;
  }

  void update( int index )  {
    counts[ index ]++;
    if( ++sumCounts >= maxSum ) {
      sumCounts = 0;
      for( auto& cnt : counts )   // rounding up !
        sumCounts += ( cnt = ( cnt + 1 ) >> 1 );
    }
  }

private:
  const int    V;             // number of bits for pmf
  const int    maxSum;        // maximum sum of counts
  int          sumCounts;     // sum of all counts
  vector<int>  counts;        // counts for letters
};
```

Arithmetic Coding with Conditional Pmfs

Basic Idea

- Switch pmf after encoding of each symbol
- Switch pmf after decoding of each symbol
- Can be combined with adaptive pmfs

Arithmetic Coding with Conditional Pmfs

Basic Idea

- Switch pmf after encoding of each symbol
- Switch pmf after decoding of each symbol
- Can be combined with adaptive pmfs

Conditions for 1D Signals (e.g., audio)

- Directly preceding sample
- Two (or more) directly preceding samples
- Function of one or more preceding samples

Arithmetic Coding with Conditional Pmfs

Basic Idea

- Switch pmf after encoding of each symbol
- Switch pmf after decoding of each symbol
- Can be combined with adaptive pmfs

Conditions for 1D Signals (e.g., audio)

- Directly preceding sample
- Two (or more) directly preceding samples
- Function of one or more preceding samples

Conditions for 2D Signals (e.g., images)

- One already coded neighboring sample
- Two or more already coded neighboring samples
- Function of already coded neighboring samples

Arithmetic Coding with Conditional Pmfs

Basic Idea

- Switch pmf after encoding of each symbol
- Switch pmf after decoding of each symbol
- Can be combined with adaptive pmfs

Conditions for 1D Signals (e.g., audio)

- Directly preceding sample
- Two (or more) directly preceding samples
- Function of one or more preceding samples

Conditions for 2D Signals (e.g., images)

- One already coded neighboring sample
- Two or more already coded neighboring samples
- Function of already coded neighboring samples

```
void encodeMessage( const vector<int>& message, ... )
{
  vector<Pmf>  pmfs( numLetters, {...} );
  ArithEncoder aenc( ... );
  int lastSymbol = 0;
  for( const auto& currSymbol : message )  {
    Pmf& currPmf   = pmfs[ lastSymbol ];
    aenc.encode    ( currSymbol, currPmf );
    currPmf.update( currSymbol );
    lastSymbol     = currSymbol;
  }
  aenc.terminate();
}
```

Arithmetic Coding with Conditional Pmfs

Basic Idea

- Switch pmf after encoding of each symbol
- Switch pmf after decoding of each symbol
- Can be combined with adaptive pmfs

Conditions for 1D Signals (e.g., audio)

- Directly preceding sample
- Two (or more) directly preceding samples
- Function of one or more preceding samples

Conditions for 2D Signals (e.g., images)

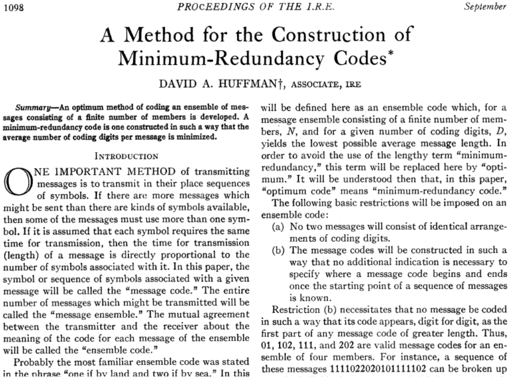
- One already coded neighboring sample
- Two or more already coded neighboring samples
- Function of already coded neighboring samples

```
void encodeMessage( const vector<int>& message, ... )
{
  vector<Pmf>  pmfs( numLetters, {...} );
  ArithEncoder aenc( ... );
  int lastSymbol = 0;
  for( const auto& currSymbol : message )  {
    Pmf& currPmf   = pmfs[ lastSymbol ];
    aenc.encode    ( currSymbol, currPmf );
    currPmf.update( currSymbol );
    lastSymbol     = currSymbol;
  }
  aenc.terminate();
}
```

```
vector<int> decodeMessage( int numSymbols, ... )
{
  vector<int>  message;
  vector<Pmf>  pmfs( numLetters, {...} );
  ArithDecoder adec( ... );
  int symbol = 0;
  while( message.size() < numSymbols )  {
    Pmf& currPmf   = pmfs[ symbol ];
    symbol         = adec.decode( currPmf );
    currPmf.update    ( symbol );
    message.push_back( symbol );
  }
  return message;
}
```

Example: Conditional Arithmetic Coding for Document Scans

Original:	1 052 713	bytes	
gzip:	183 705	bytes	(17.45 %)
bzip2:	169 049	bytes	(16.05 %)
lzip:	140 307	bytes	(13.33 %)



1098 PROCEEDINGS OF THE I.R.E. September

A Method for the Construction of Minimum-Redundancy Codes*

DAVID A. HUFFMAN†, ASSOCIATE, IRE

Summary—**An optimum method of coding an ensemble of messages consisting of a finite number of members is developed. A minimum-redundancy code is one constructed in such a way that the average number of coding digits per message is minimized.**

INTRODUCTION

ONE IMPORTANT METHOD of transmitting messages is to transmit in their place sequences of symbols. If there are more messages which might be sent than there are kinds of symbols available, then some of the messages must use more than one symbol. If it is assumed that each symbol requires the same time for transmission, then the time for transmission (length) of a message is directly proportional to the number of symbols associated with it. In this paper, the symbol or sequence of symbols associated with a given message will be called the "message code." The entire number of messages which might be transmitted will be called the "message ensemble." The mutual agreement between the transmitter and the receiver about the meaning of the code for each message of the ensemble will be called the "ensemble code."

Probably the most familiar ensemble code was stated in the phrase "one if by land and two if by sea." In this will be defined here as an ensemble code which, for a message ensemble consisting of a finite number of members, N, and for a given number of coding digits, D, yields the lowest possible average message length. In order to avoid the use of the lengthy term "minimum-redundancy," this term will be replaced here by "optimum." It will be understood then that, in this paper, "optimum code" means "minimum-redundancy code."

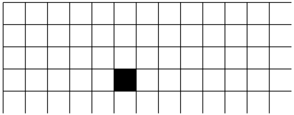
The following basic restrictions will be imposed on an ensemble code:

(a) No two messages will consist of identical arrangements of coding digits.

(b) The message codes will be constructed in such a way that no additional indication is necessary to specify where a message code begins and ends once the starting point of a sequence of messages is known.

Restriction (b) necessitates that no message be coded in such a way that its code appears, digit for digit, as the first part of any message code of greater length. Thus, 01, 102, 111, and 202 are valid message codes for an ensemble of four members. For instance, a sequence of these messages 1111022020101111102 can be broken up

Example: Conditional Arithmetic Coding for Document Scans



Original:	1 052 713	bytes	
gzip:	183 705	bytes	(17.45 %)
bzip2:	169 049	bytes	(16.05 %)
lzip:	140 307	bytes	(13.33 %)

1. **No condition**: 1 binary pmf
   ➡ File size = 418 369 bytes (39.74 %)

1098 *PROCEEDINGS OF THE I.R.E.* *September*

A Method for the Construction of Minimum-Redundancy Codes*

DAVID A. HUFFMAN†, ASSOCIATE, IRE

***Summary*—An optimum method of coding an ensemble of messages consisting of a finite number of members is developed. A minimum-redundancy code is one constructed in such a way that the average number of coding digits per message is minimized.**

Introduction

ONE IMPORTANT METHOD of transmitting messages is to transmit in their place sequences of symbols. If there are more messages which might be sent than there are kinds of symbols available, then some of the messages must use more than one symbol. If it is assumed that each symbol requires the same time for transmission, then the time for transmission (length) of a message is directly proportional to the number of symbols associated with it. In this paper, the symbol or sequence of symbols associated with a given message will be called the "message code." The entire number of messages which might be transmitted will be called the "message ensemble." The mutual agreement between the transmitter and the receiver about the meaning of the code for each message of the ensemble will be called the "ensemble code."

Probably the most familiar ensemble code was stated in the phrase "one if by land and two if by sea." In this

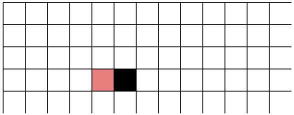
will be defined here as an ensemble code which, for a message ensemble consisting of a finite number of members, N, and for a given number of coding digits, D, yields the lowest possible average message length. In order to avoid the use of the lengthy term "minimum-redundancy," this term will be replaced here by "optimum." It will be understood then that, in this paper, "optimum code" means "minimum-redundancy code."

The following basic restrictions will be imposed on an ensemble code:

(a) No two messages will consist of identical arrangements of coding digits.
(b) The message codes will be constructed in such a way that no additional indication is necessary to specify where a message code begins and ends once the starting point of a sequence of messages is known.

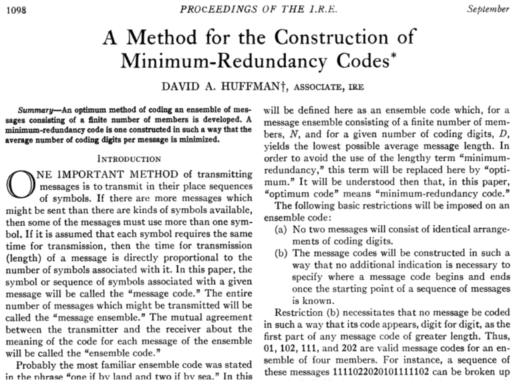
Restriction (b) necessitates that no message be coded in such a way that its code appears, digit for digit, as the first part of any message code of greater length. Thus, 01, 102, 111, and 202 are valid message codes for an ensemble of four members. For instance, a sequence of these messages 1111022020101111102 can be broken up

Example: Conditional Arithmetic Coding for Document Scans



Original:	1 052 713	bytes	
gzip:	183 705	bytes	(17.45 %)
bzip2:	169 049	bytes	(16.05 %)
lzip:	140 307	bytes	(13.33 %)

1. **No condition**: 1 binary pmf
   ➡ File size = 418 369 bytes (39.74 %)
2. **Left neighbour**: 2 binary pmfs
   ➡ File size = 192 841 bytes (18.32 %)



1098 PROCEEDINGS OF THE I.R.E. September

A Method for the Construction of Minimum-Redundancy Codes*

DAVID A. HUFFMAN†, ASSOCIATE, IRE

Summary—An optimum method of coding an ensemble of messages consisting of a finite number of members is developed. A minimum-redundancy code is one constructed in such a way that the average number of coding digits per message is minimized.

INTRODUCTION

ONE IMPORTANT METHOD of transmitting messages is to transmit in their place sequences of symbols. If there are more messages which might be sent than there are kinds of symbols available, then some of the messages must use more than one symbol. If it is assumed that each symbol requires the same time for transmission, then the time for transmission (length) of a message is directly proportional to the number of symbols associated with it. In this paper, the symbol or sequence of symbols associated with a given message will be called the "message code." The entire number of messages which might be transmitted will be called the "message ensemble." The mutual agreement between the transmitter and the receiver about the meaning of the code for each message of the ensemble will be called the "ensemble code."

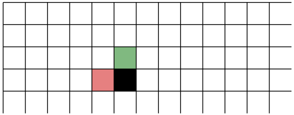
Probably the most familiar ensemble code was stated in the phrase "one if by land and two if by sea." In this will be defined here as an ensemble code which, for a message ensemble consisting of a finite number of members, N, and for a given number of coding digits, D, yields the lowest possible average message length. In order to avoid the use of the lengthy term "minimum-redundancy," this term will be replaced here by "optimum." It will be understood then that, in this paper, "optimum code" means "minimum-redundancy code."

The following basic restrictions will be imposed on an ensemble code:

(a) No two messages will consist of identical arrangements of coding digits.
(b) The message codes will be constructed in such a way that no additional indication is necessary to specify where a message code begins and ends once the starting point of a sequence of messages is known.

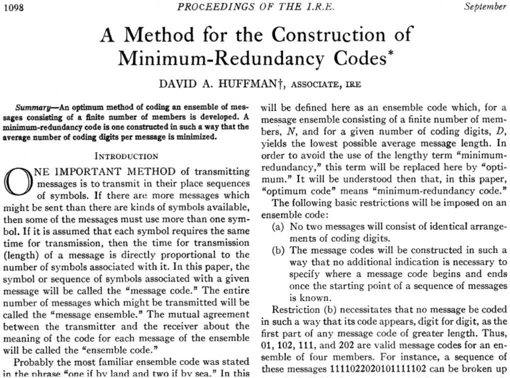
Restriction (b) necessitates that no message be coded in such a way that its code appears, digit for digit, as the first part of any message code of greater length. Thus, 01, 102, 111, and 202 are valid message codes for an ensemble of four members. For instance, a sequence of these messages 111022020101111102 can be broken up

Example: Conditional Arithmetic Coding for Document Scans



Original:	1 052 713	bytes	
gzip:	183 705	bytes	(17.45 %)
bzip2:	169 049	bytes	(16.05 %)
lzip:	140 307	bytes	(13.33 %)

1. **No condition**: 1 binary pmf
   ➡ File size = 418 369 bytes (39.74 %)
2. **Left neighbour**: 2 binary pmfs
   ➡ File size = 192 841 bytes (18.32 %)
3. **Left and above**: 4 binary pmfs
   ➡ File size = 120 198 bytes (11.42 %)



1098 *PROCEEDINGS OF THE I.R.E.* *September*

A Method for the Construction of Minimum-Redundancy Codes*

DAVID A. HUFFMAN†, ASSOCIATE, IRE

Summary—An optimum method of coding an ensemble of messages consisting of a finite number of members is developed. A minimum-redundancy code is one constructed in such a way that the average number of coding digits per message is minimized.

INTRODUCTION

ONE IMPORTANT METHOD of transmitting messages is to transmit in their place sequences of symbols. If there are more messages which might be sent than there are kinds of symbols available, then some of the messages must use more than one symbol. If it is assumed that each symbol requires the same time for transmission, then the time for transmission (length) of a message is directly proportional to the number of symbols associated with it. In this paper, the symbol or sequence of symbols associated with a given message will be called the "message code." The entire number of messages which might be transmitted will be called the "message ensemble." The mutual agreement between the transmitter and the receiver about the meaning of the code for each message of the ensemble will be called the "ensemble code."

Probably the most familiar ensemble code was stated in the phrase "one if by land and two if by sea." In this

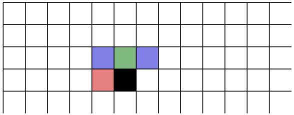
will be defined here as an ensemble code which, for a message ensemble consisting of a finite number of members, N, and for a given number of coding digits, D, yields the lowest possible average message length. In order to avoid the use of the lengthy term "minimum-redundancy," this term will be replaced here by "optimum." It will be understood then that, in this paper, "optimum code" means "minimum-redundancy code."

The following basic restrictions will be imposed on an ensemble code:

(a) No two messages will consist of identical arrangements of coding digits.
(b) The message codes will be constructed in such a way that no additional indication is necessary to specify where a message code begins and ends once the starting point of a sequence of messages is known.

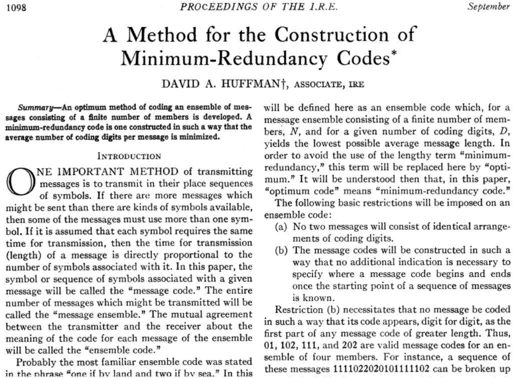
Restriction (b) necessitates that no message be coded in such a way that its code appears, digit for digit, as the first part of any message code of greater length. Thus, 01, 102, 111, and 202 are valid message codes for an ensemble of four members. For instance, a sequence of these messages 1111022020101111102 can be broken up

Example: Conditional Arithmetic Coding for Document Scans



Original:	1 052 713	bytes	
gzip:	183 705	bytes	(17.45 %)
bzip2:	169 049	bytes	(16.05 %)
lzip:	140 307	bytes	(13.33 %)

1. **No condition**: 1 binary pmf
   ➡ File size = 418 369 bytes (39.74 %)
2. **Left neighbour**: 2 binary pmfs
   ➡ File size = 192 841 bytes (18.32 %)
3. **Left and above**: 4 binary pmfs
   ➡ File size = 120 198 bytes (11.42 %)
4. **Four neighbours**: 16 binary pmfs
   ➡ File size = 101 819 bytes (9.67 %)



1098 PROCEEDINGS OF THE I.R.E. September

A Method for the Construction of Minimum-Redundancy Codes*

DAVID A. HUFFMAN†, ASSOCIATE, IRE

Summary—An optimum method of coding an ensemble of messages consisting of a finite number of members is developed. A minimum-redundancy code is one constructed in such a way that the average number of coding digits per message is minimized.

INTRODUCTION

ONE IMPORTANT METHOD of transmitting messages is to transmit in their place sequences of symbols. If there are more messages which might be sent than there are kinds of symbols available, then some of the messages must use more than one symbol. If it is assumed that each symbol requires the same time for transmission, then the time for transmission (length) of a message is directly proportional to the number of symbols associated with it. In this paper, the symbol or sequence of symbols associated with a given message will be called the "message code." The entire number of messages which might be transmitted will be called the "message ensemble." The mutual agreement between the transmitter and the receiver about the meaning of the code for each message of the ensemble will be called the "ensemble code."

Probably the most familiar ensemble code was stated in the phrase "one if by land and two if by sea." In this

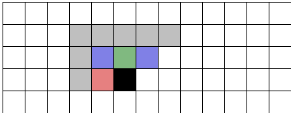
will be defined here as an ensemble code which, for a message ensemble consisting of a finite number of members, N, and for a given number of coding digits, D, yields the lowest possible average message length. In order to avoid the use of the lengthy term "minimum-redundancy," this term will be replaced here by "optimum." It will be understood then that, in this paper, "optimum code" means "minimum-redundancy code."

The following basic restrictions will be imposed on an ensemble code:

(a) No two messages will consist of identical arrangements of coding digits.
(b) The message codes will be constructed in such a way that no additional indication is necessary to specify where a message code begins and ends once the starting point of a sequence of messages is known.

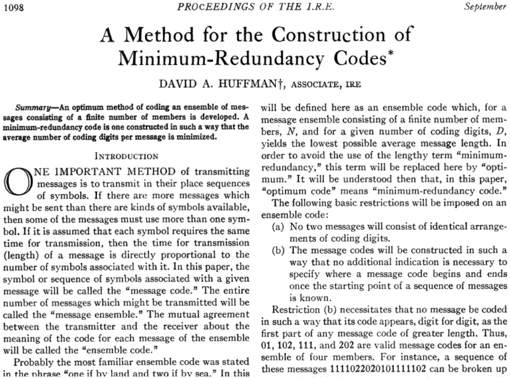
Restriction (b) necessitates that no message be coded in such a way that its code appears, digit for digit, as the first part of any message code of greater length. Thus, 01, 102, 111, and 202 are valid message codes for an ensemble of four members. For instance, a sequence of these messages 1111022020101111102 can be broken up

Example: Conditional Arithmetic Coding for Document Scans



Original:	1 052 713	bytes	
gzip:	183 705	bytes	(17.45 %)
bzip2:	169 049	bytes	(16.05 %)
lzip:	140 307	bytes	(13.33 %)

1. **No condition**: 1 binary pmf
   ➡ File size = 418 369 bytes (39.74 %)
2. **Left neighbour**: 2 binary pmfs
   ➡ File size = 192 841 bytes (18.32 %)
3. **Left and above**: 4 binary pmfs
   ➡ File size = 120 198 bytes (11.42 %)
4. **Four neighbours**: 16 binary pmfs
   ➡ File size = 101 819 bytes (9.67 %)
5. **Eleven neighbours**: 2048 binary pmfs
   ➡ File size = 92 527 bytes (8.79 %)



1098 PROCEEDINGS OF THE I.R.E. September

A Method for the Construction of Minimum-Redundancy Codes*

DAVID A. HUFFMAN†, ASSOCIATE, IRE

Summary—An optimum method of coding an ensemble of messages consisting of a finite number of members is developed. A minimum-redundancy code is one constructed in such a way that the average number of coding digits per message is minimized.

INTRODUCTION

ONE IMPORTANT METHOD of transmitting messages is to transmit in their place sequences of symbols. If there are more messages which might be sent than there are kinds of symbols available, then some of the messages must use more than one symbol. If it is assumed that each symbol requires the same time for transmission, then the time for transmission (length) of a message is directly proportional to the number of symbols associated with it. In this paper, the symbol or sequence of symbols associated with a given message will be called the "message code." The entire number of messages which might be transmitted will be called the "message ensemble." The mutual agreement between the transmitter and the receiver about the meaning of the code for each message of the ensemble will be called the "ensemble code."

Probably the most familiar ensemble code was stated in the phrase "one if by land and two if by sea." In this

will be defined here as an ensemble code which, for a message ensemble consisting of a finite number of members, N, and for a given number of coding digits, D, yields the lowest possible average message length. In order to avoid the use of the lengthy term "minimum-redundancy," this term will be replaced here by "optimum." It will be understood then that, in this paper, "optimum code" means "minimum-redundancy code."

The following basic restrictions will be imposed on an ensemble code:

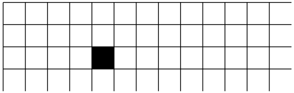
(a) No two messages will consist of identical arrangements of coding digits.
(b) The message codes will be constructed in such a way that no additional indication is necessary to specify where a message code begins and ends once the starting point of a sequence of messages is known.

Restriction (b) necessitates that no message be coded in such a way that its code appears, digit for digit, as the first part of any message code of greater length. Thus, 01, 102, 111, and 202 are valid message codes for an ensemble of four members. For instance, a sequence of these messages 1111022020101111102 can be broken up

Example: Conditional Arithmetic Coding for 8-bit Images

Original:	262 159	bytes	(512×512)
gzip:	222 999	bytes	(85.06 %)
bzip2:	173 877	bytes	(66.33 %)
lzip:	180 000	bytes	(68.66 %)

Example: Conditional Arithmetic Coding for 8-bit Images

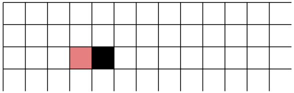


Original:	262 159	bytes	(512×512)
gzip:	222 999	bytes	(85.06 %)
bzip2:	173 877	bytes	(66.33 %)
lzip:	180 000	bytes	(68.66 %)

1 No condition:

- ➡ 256 probability masses (2^8)
- ➡ File size = 240 112 bytes (91.59 %)

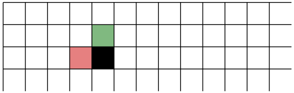
Example: Conditional Arithmetic Coding for 8-bit Images



1. **No condition**:
   - ➡ 256 probability masses (2^8)
   - ➡ File size = 240 112 bytes (91.59 %)
2. **Left sample**:
   - ➡ 65 536 probability masses ($2^8 \cdot 2^8$)
   - ➡ File size = 179 179 bytes (68.35 %)

Original:	262 159	bytes	(512×512)
gzip:	222 999	bytes	(85.06 %)
bzip2:	173 877	bytes	(66.33 %)
lzip:	180 000	bytes	(68.66 %)

Example: Conditional Arithmetic Coding for 8-bit Images



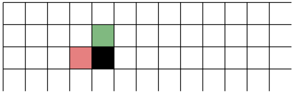
1 No condition:
- ➜ 256 probability masses (2^8)
- ➜ File size = 240 112 bytes (91.59 %)

2 Left sample:
- ➜ 65 536 probability masses ($2^8 \cdot 2^8$)
- ➜ File size = 179 179 bytes (68.35 %)

3 Left sample and above sample:
- ➜ 16 777 216 probability masses ($2^8 \cdot 2^8 \cdot 2^8$)
- ➜ File size = 221 849 bytes (84.62 %)

Original:	262 159	bytes	(512 × 512)
gzip:	222 999	bytes	(85.06 %)
bzip2:	173 877	bytes	(66.33 %)
lzip:	180 000	bytes	(68.66 %)

Example: Conditional Arithmetic Coding for 8-bit Images



Original:	262 159	bytes	(512×512)
gzip:	222 999	bytes	(85.06 %)
bzip2:	173 877	bytes	(66.33 %)
lzip:	180 000	bytes	(68.66 %)

1 No condition:
- ➡ 256 probability masses (2^8)
- ➡ File size = 240 112 bytes (91.59 %)

2 Left sample:
- ➡ 65 536 probability masses ($2^8 \cdot 2^8$)
- ➡ File size = 179 179 bytes (68.35 %)

3 Left sample and above sample:
- ➡ 16 777 216 probability masses ($2^8 \cdot 2^8 \cdot 2^8$)
- ➡ File size = 221 849 bytes (84.62 %)

- ➡ **Too many probability masses**
- ➡ **Pmfs do not adapt to image statistics**

Example: Conditional Arithmetic Coding for 16-bit Audio

Original:	26 559 960	bytes	(5:01 minutes)
gzip:	24 926 843	bytes	(93.85 %)
bzip2:	22 445 509	bytes	(84.51 %)
lzip:	23 777 258	bytes	(89.52 %)

Example: Conditional Arithmetic Coding for 16-bit Audio

Original:	26 559 960	bytes	(5:01 minutes)
gzip:	24 926 843	bytes	(93.85 %)
bzip2:	22 445 509	bytes	(84.51 %)
lzip:	23 777 258	bytes	(89.52 %)

1 No condition:

- ➡ 65 536 probability masses (2^{16})
- ➡ File size = 23 700 606 bytes (89.23 %)

Example: Conditional Arithmetic Coding for 16-bit Audio

Original:	26 559 960	bytes	(5:01 minutes)
gzip:	24 926 843	bytes	(93.85 %)
bzip2:	22 445 509	bytes	(84.51 %)
lzip:	23 777 258	bytes	(89.52 %)

1 **No condition**:
- ➡ 65 536 probability masses (2^{16})
- ➡ File size = 23 700 606 bytes (89.23 %)

2 **Previous sample**:
- ➡ 4 294 967 296 probability masses ($2^{16} \cdot 2^{16}$)

Example: Conditional Arithmetic Coding for 16-bit Audio

Original:	26 559 960	bytes	(5:01 minutes)
gzip:	24 926 843	bytes	(93.85 %)
bzip2:	22 445 509	bytes	(84.51 %)
lzip:	23 777 258	bytes	(89.52 %)

1 No condition:
- ➡ 65 536 probability masses (2^{16})
- ➡ File size = 23 700 606 bytes (89.23 %)

2 Previous sample:
- ➡ 4 294 967 296 probability masses ($2^{16} \cdot 2^{16}$)
- ➡ **Not possible on normal computers**
- ➡ **Requires 16 GByte of memory (when we use 32 bit per probability)**
- ➡ **Would not adapt well to statistics**

Example: Conditional Arithmetic Coding for 16-bit Audio

Original:	26 559 960	bytes	(5:01 minutes)
gzip:	24 926 843	bytes	(93.85 %)
bzip2:	22 445 509	bytes	(84.51 %)
lzip:	23 777 258	bytes	(89.52 %)

1 No condition:

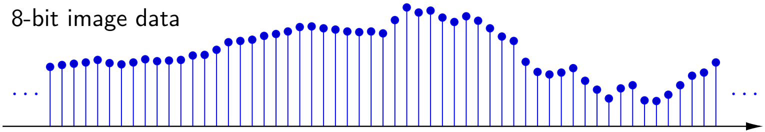
- ➡ 65 536 probability masses (2^{16})
- ➡ File size = 23 700 606 bytes (89.23 %)

2 Previous sample:

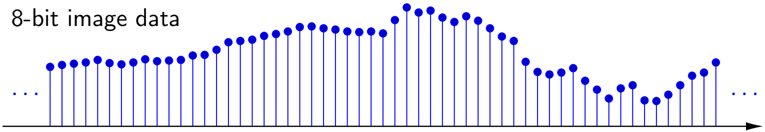
- ➡ 4 294 967 296 probability masses ($2^{16} \cdot 2^{16}$)
- ➡ **Not possible on normal computers**
- ➡ **Requires 16 GByte of memory (when we use 32 bit per probability)**
- ➡ **Would not adapt well to statistics**

➡ **Cannot exploit dependencies between symbols using this type of coding !!!**

Analysis: Typical Properties of Real Signals

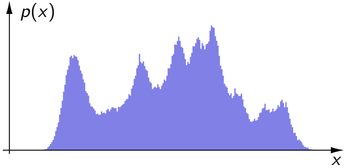


Analysis: Typical Properties of Real Signals

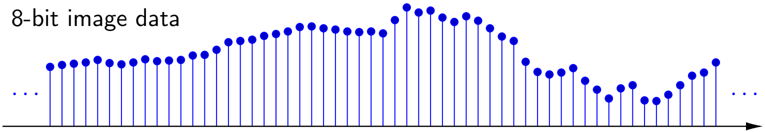


Marginal Pmf

- Not much room for compression
- Entropy $H = 7.45$ (8-bit data)

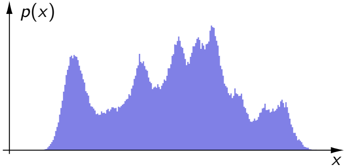


Analysis: Typical Properties of Real Signals



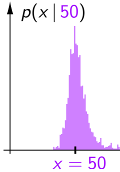
Marginal Pmf

- Not much room for compression
- Entropy $H = 7.45$ (8-bit data)

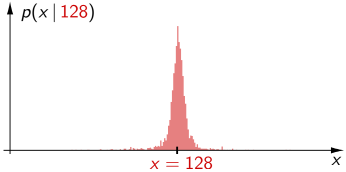


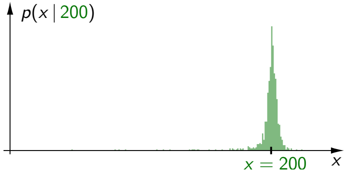
Conditional Pmfs

- Significantly narrower than marginal pmf (room for compression)
- Shape is very similar for all conditions, but shifted by x_{n-1}



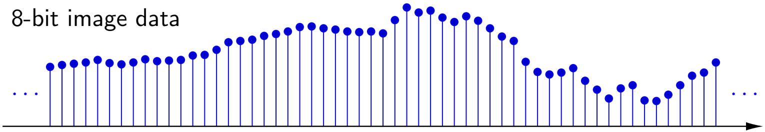
$$p(x \mid a) = \mathrm{P}(X_n = x \mid X_{n-1} = a)$$

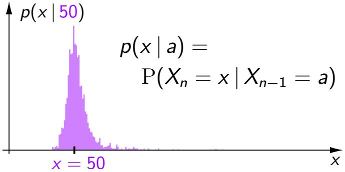




Analysis: Typical Properties of Real Signals

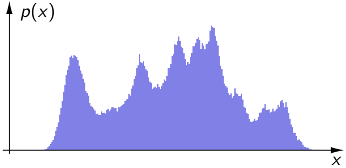
8-bit image data

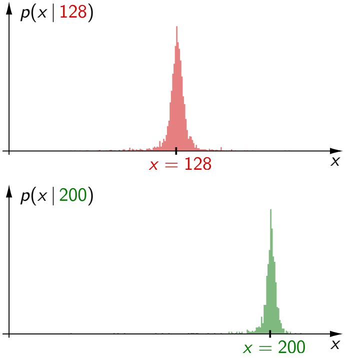




Marginal Pmf

- Not much room for compression
- Entropy $H = 7.45$ (8-bit data)



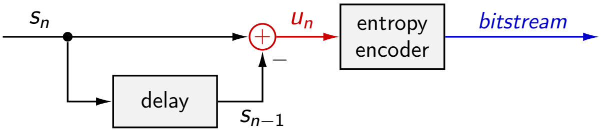


Conditional Pmfs

- Significantly narrower than marginal pmf (room for compression)
- Shape is very similar for all conditions, but shifted by x_{n-1}

➡ **Idea: Code difference to previous sample !!!**

Simple Predictive Coding using Preceding Sample



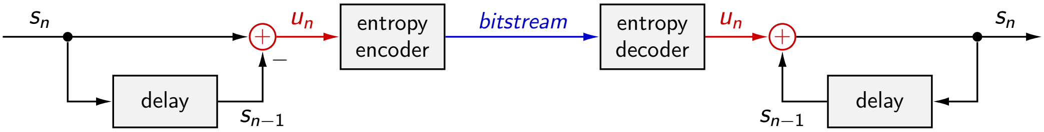
Encoding

- Generate difference sample

$$u_n = s_n - s_{n-1}$$

- Encode difference sample u_n

Simple Predictive Coding using Preceding Sample



Encoding

- Generate difference sample

$$u_n = s_n - s_{n-1}$$

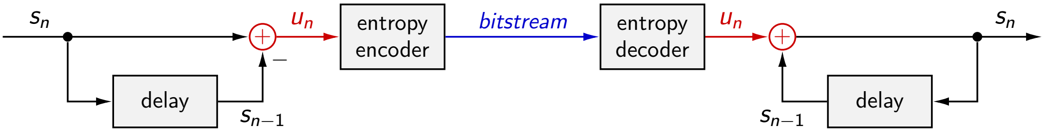
- Encode difference sample u_n

Decoding

- Decode difference sample u_n
- Reconstruct original sample

$$s_n = s_{n-1} + u_n$$

Simple Predictive Coding using Preceding Sample



Encoding

- Generate difference sample

$$u_n = s_n - s_{n-1}$$

- Encode difference sample u_n

Decoding

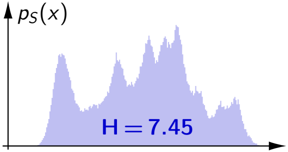
- Decode difference sample u_n
- Reconstruct original sample

$$s_n = s_{n-1} + u_n$$

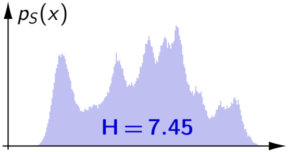
Basic Effect

- Prediction removes large part of inter-symbol dependencies before actual entropy coding
- Reduces required complexity for the entropy coding (e.g., marginal instead of conditional coding)

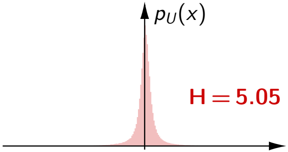
How Well Does That Type of Prediction Work ?

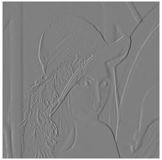


How Well Does That Type of Prediction Work ?

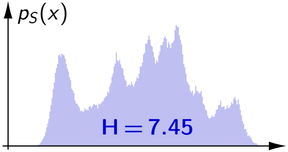


$$u_n = s_n - s_{n-1}$$

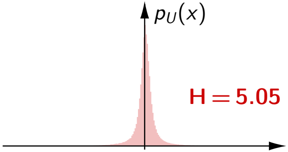


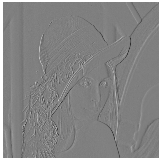


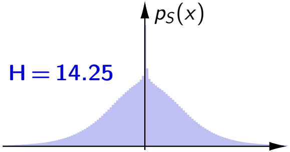
How Well Does That Type of Prediction Work ?



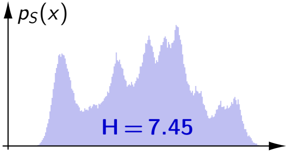
$$u_n = s_n - s_{n-1} \longrightarrow$$



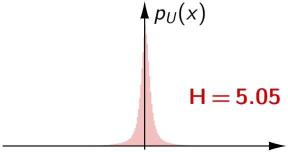


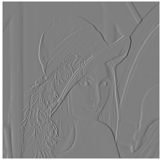


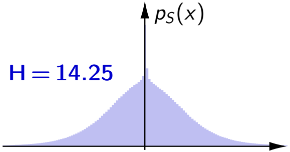
How Well Does That Type of Prediction Work ?



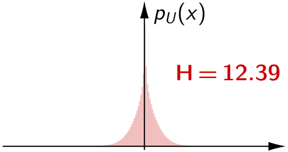
$u_n = s_n - s_{n-1}$





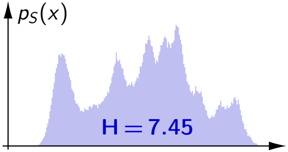


$u_n = s_n - s_{n-1}$

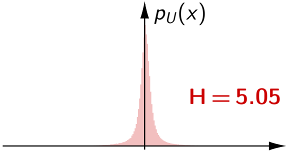


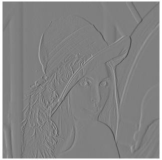
saves nearly 2 bits per sample !

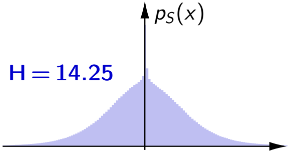
How Well Does That Type of Prediction Work ?



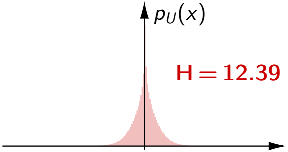
$u_n = s_n - s_{n-1}$





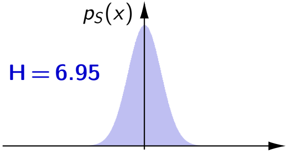


$u_n = s_n - s_{n-1}$

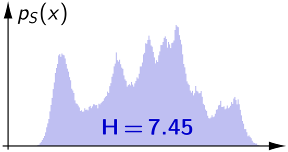


saves nearly 2 bits per sample !

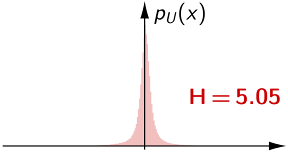
White Gaussian Noise

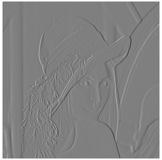


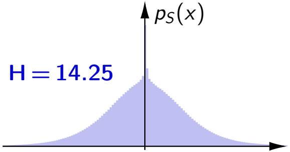
How Well Does That Type of Prediction Work ?



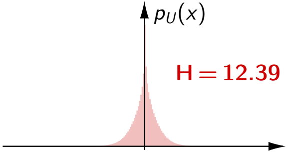
$u_n = s_n - s_{n-1}$





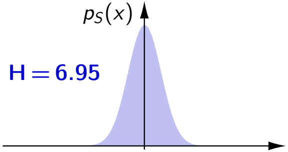


$u_n = s_n - s_{n-1}$

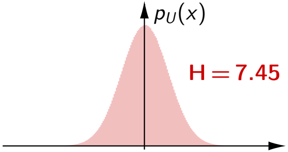


saves nearly 2 bits per sample !

White Gaussian Noise

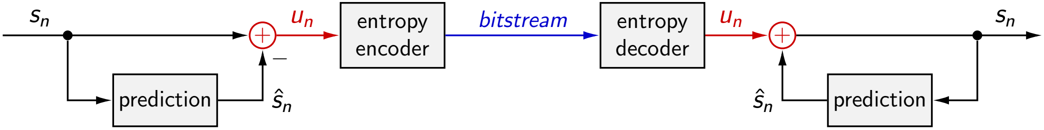


$u_n = s_n - s_{n-1}$



prediction increases entropy !

General Predictive Coding



Predictive Lossless Coding

- Predict current sample s_n using a function of preceding samples

$$\hat{s}_n = f(s_{n-1}, s_{n-2}, \cdots)$$

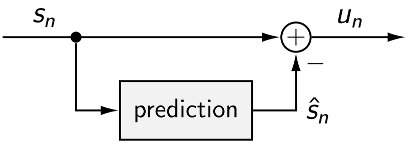
- Entropy coding of prediction error samples

$$u_n = s_n - \hat{s}_n$$

- Decoder uses exactly the same prediction and reconstructs the original samples according to

$$s_n = \hat{s}_n + u_n$$

Choice of Observation Set and Predictor

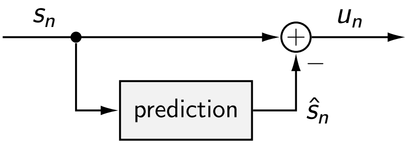


Choice of Observation Set $\mathcal{B}_n$

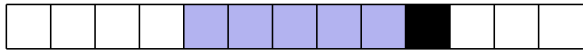
- Use small number of preceding samples for prediction

➡ Choose the samples $\mathcal{B}_n$ with highest dependencies to current sample s_n

Choice of Observation Set and Predictor



1D signals:



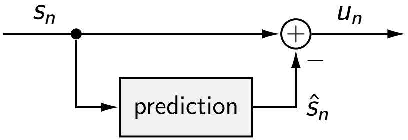
Choice of Observation Set $\mathcal{B}_n$

- Use small number of preceding samples for prediction

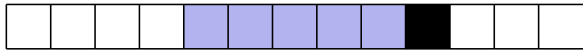
➡ Choose the samples $\mathcal{B}_n$ with highest dependencies to current sample s_n

  - 1D signals (e.g., audio): N directly preceding samples $\rightarrow$ $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots, s_{n-N}\}$

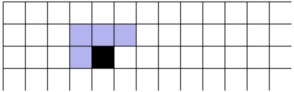
Choice of Observation Set and Predictor



1D signals:



2D signals:



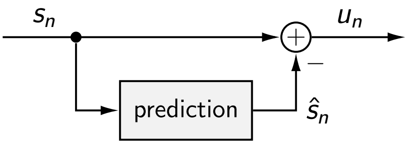
Choice of Observation Set $\mathcal{B}_n$

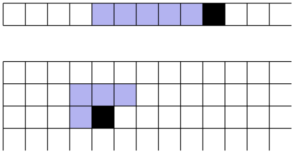
- Use small number of preceding samples for prediction

➡ Choose the samples $\mathcal{B}_n$ with highest dependencies to current sample s_n

- 1D signals (e.g., audio): N directly preceding samples $\rightarrow$ $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots, s_{n-N}\}$
- 2D signals (e.g., images): Samples in causal direct neighborhood

Choice of Observation Set and Predictor





Choice of Observation Set $\mathcal{B}_n$

- Use small number of preceding samples for prediction

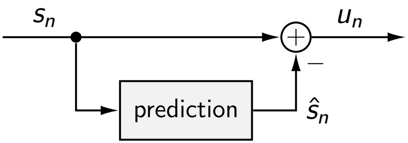
➡ Choose the samples $\mathcal{B}_n$ with highest dependencies to current sample s_n

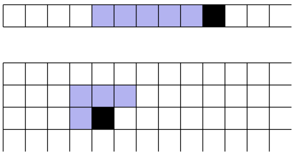
- 1D signals (e.g., audio): N directly preceding samples $\rightarrow$ $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots, s_{n-N}\}$
- 2D signals (e.g., images): Samples in causal direct neighborhood

Choice of Predictor for given Observation Set

- Question: What function $\hat{s}_n = f(\mathcal{B}_n)$ should we use for prediction ?

Choice of Observation Set and Predictor





Choice of Observation Set $\mathcal{B}_n$

- Use small number of preceding samples for prediction

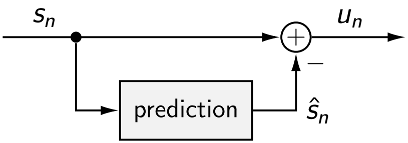
➡ Choose the samples $\mathcal{B}_n$ with highest dependencies to current sample s_n

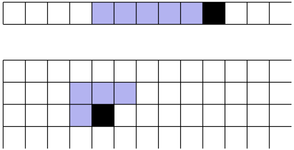
- 1D signals (e.g., audio): N directly preceding samples $\rightarrow$ $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots, s_{n-N}\}$
- 2D signals (e.g., images): Samples in causal direct neighborhood

Choice of Predictor for given Observation Set

- Question: What function $\hat{s}_n = f(\mathcal{B}_n)$ should we use for prediction ?
- Want to minimize entropy $H(U_n)$ of prediction error samples $\{u_n\}$ (difficult to access directly)

Choice of Observation Set and Predictor





Choice of Observation Set $\mathcal{B}_n$

- Use small number of preceding samples for prediction

➡ Choose the samples $\mathcal{B}_n$ with highest dependencies to current sample s_n

  - 1D signals (e.g., audio): N directly preceding samples $\rightarrow$ $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots, s_{n-N}\}$
  - 2D signals (e.g., images): Samples in causal direct neighborhood

Choice of Predictor for given Observation Set

- Question: What function $\hat{s}_n = f(\mathcal{B}_n)$ should we use for prediction ?
- Want to minimize entropy $H(U_n)$ of prediction error samples $\{u_n\}$ (difficult to access directly)

➡ Can minimize variance σ_U^2 and ignore shape of pmf $p(u)$

Optimization Criterion for Deriving Predictor

Typical Optimization Criterion

- Minimize Energy of Prediction Error Signal

$$\varepsilon_U^2 = \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\left\{ \left(S_n - \hat{S}_n \right)^2 \right\} = \mathrm{E}\left\{ \left(S_n - f(\mathcal{B}_n) \right)^2 \right\}$$

Optimization Criterion for Deriving Predictor

Typical Optimization Criterion

- Minimize Energy of Prediction Error Signal

$$\varepsilon_U^2 = \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\Big\{ \Big(S_n - \hat{S}_n \Big)^2 \Big\} = \mathrm{E}\Big\{ \Big(S_n - f(\mathcal{B}_n) \Big)^2 \Big\}$$

- Reformulate prediction error energy

$$\begin{aligned} \varepsilon_U^2 &= \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\Big\{ (U_n - \mu_U + \mu_u)^2 \Big\} \\ &= \mathrm{E}\Big\{ (U_n - \mu_U)^2 \Big\} + \mu_U^2 + 2\,\mu_U\,\mathrm{E}\{\, U_n - \mu_U \,\} \\ &= \sigma_U^2 + \mu_U^2 \end{aligned}$$

Optimization Criterion for Deriving Predictor

Typical Optimization Criterion

- Minimize Energy of Prediction Error Signal

$$\varepsilon_U^2 = \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\left\{ \left(S_n - \hat{S}_n \right)^2 \right\} = \mathrm{E}\left\{ \left(S_n - f(\mathcal{B}_n) \right)^2 \right\}$$

- Reformulate prediction error energy

$$\begin{aligned} \varepsilon_U^2 &= \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\left\{ (U_n - \mu_U + \mu_u)^2 \right\} \\ &= \mathrm{E}\left\{ (U_n - \mu_U)^2 \right\} + \mu_U^2 + 2\,\mu_U\,\mathrm{E}\{\, U_n - \mu_U \,\} \\ &= \sigma_U^2 + \mu_U^2 \end{aligned}$$

➡ Minimization of squared prediction error ε_U^2 implies minimization of variance σ_U^2 and mean μ_U^2

Optimization Criterion for Deriving Predictor

Typical Optimization Criterion

- Minimize Energy of Prediction Error Signal

$$\varepsilon_U^2 = \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\Big\{ \Big(S_n - \hat{S}_n \Big)^2 \Big\} = \mathrm{E}\Big\{ \Big(S_n - f(\mathcal{B}_n) \Big)^2 \Big\}$$

- Reformulate prediction error energy

$$\begin{aligned} \varepsilon_U^2 &= \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\Big\{ (U_n - \mu_U + \mu_u)^2 \Big\} \\ &= \mathrm{E}\Big\{ (U_n - \mu_U)^2 \Big\} + \mu_U^2 + 2\,\mu_U\,\mathrm{E}\{\, U_n - \mu_U \,\} \\ &= \sigma_U^2 + \mu_U^2 \end{aligned}$$

➡ Minimization of squared prediction error ε_U^2 implies minimization of variance σ_U^2 and mean μ_U^2

➡ Minimize the width of the pmf $p_U(u)$, but ignore its actual shape

Optimization Criterion for Deriving Predictor

Typical Optimization Criterion

- Minimize Energy of Prediction Error Signal

$$\varepsilon_U^2 = \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\Big\{ \big(S_n - \hat{S}_n \big)^2 \Big\} = \mathrm{E}\Big\{ \big(S_n - f(\mathcal{B}_n) \big)^2 \Big\}$$

- Reformulate prediction error energy

$$\begin{aligned}
\varepsilon_U^2 &= \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\Big\{ (U_n - \mu_U + \mu_u)^2 \Big\} \\
&= \mathrm{E}\Big\{ (U_n - \mu_U)^2 \Big\} + \mu_U^2 + 2\,\mu_U\,\mathrm{E}\{\, U_n - \mu_U \,\} \\
&= \sigma_U^2 + \mu_U^2
\end{aligned}$$

➡ Minimization of squared prediction error ε_U^2 implies minimization of variance σ_U^2 and mean μ_U^2

➡ Minimize the width of the pmf $p_U(u)$, but ignore its actual shape

➡ Suitable alternative to minimization of the entropy $H(U)$

Optimal Predictor for Given Observation Set

- Question: What value a minimizes the prediction error energy ?

$$\varepsilon_U^2 = \mathrm{E}\{\,(S_n - a)^2\,\}$$

Optimal Predictor for Given Observation Set

- Question: What value *a* minimizes the prediction error energy ?

$$\varepsilon_U^2 \;=\; \mathrm{E}\{\,(S_n - a)^2\,\} = \mathrm{E}\{\,(S_n - \mathrm{E}\{\,S_n\,\} + \mathrm{E}\{\,S_n\,\} - a)^2\,\}$$

Optimal Predictor for Given Observation Set

- Question: What value a minimizes the prediction error energy ?

$$\begin{aligned} \varepsilon_U^2 &= \mathrm{E}\{\, (S_n - a)^2 \,\} = \mathrm{E}\{\, (S_n - \mathrm{E}\{\, S_n \,\} + \mathrm{E}\{\, S_n \,\} - a)^2 \,\} \\ &= \mathrm{E}\{\, (S_n - \mathrm{E}\{\, S_n \,\})^2 \,\} + \mathrm{E}\{\, (\mathrm{E}\{\, S_n \,\} - a)^2 \,\} + 2\,\mathrm{E}\Big\{\, (S_n - \mathrm{E}\{\, S_n \,\})\,(\mathrm{E}\{\, S_n \,\} - a) \,\Big\} \end{aligned}$$

Optimal Predictor for Given Observation Set

- Question: What value a minimizes the prediction error energy ?

$$
\begin{aligned}
\varepsilon_U^2 &= \mathrm{E}\{\, (S_n - a)^2 \,\} = \mathrm{E}\{\, (S_n - \mathrm{E}\{\, S_n \,\} + \mathrm{E}\{\, S_n \,\} - a)^2 \,\} \\
&= \mathrm{E}\{\, (S_n - \mathrm{E}\{\, S_n \,\})^2 \,\} + \mathrm{E}\{\, (\mathrm{E}\{\, S_n \,\} - a)^2 \,\} + 2\,\mathrm{E}\Big\{\, (S_n - \mathrm{E}\{\, S_n \,\})\,(\mathrm{E}\{\, S_n \,\} - a) \,\Big\} \\
&= \sigma_S^2 + \underbrace{(\mathrm{E}\{\, S_n \,\} - a)^2}_{\geq 0}
\end{aligned}
$$

Optimal Predictor for Given Observation Set

- Question: What value a minimizes the prediction error energy ?

$$
\begin{aligned}
\varepsilon_U^2 &= \mathrm{E}\{\,(S_n - a)^2\,\} = \mathrm{E}\{\,(S_n - \mathrm{E}\{\,S_n\,\} + \mathrm{E}\{\,S_n\,\} - a)^2\,\} \\
&= \mathrm{E}\{\,(S_n - \mathrm{E}\{\,S_n\,\})^2\,\} + \mathrm{E}\{\,(\mathrm{E}\{\,S_n\,\} - a)^2\,\} + 2\,\mathrm{E}\Big\{\,(S_n - \mathrm{E}\{\,S_n\,\})\,(\mathrm{E}\{\,S_n\,\} - a)\,\Big\} \\
&= \sigma_S^2 + \underbrace{(\mathrm{E}\{\,S_n\,\} - a)^2}_{\geq 0}
\end{aligned}
$$

➡ Prediction error energy is minimized by mean, i.e., by setting $a = \mathrm{E}\{\,S_n\,\}$

Optimal Predictor for Given Observation Set

- Question: What value a minimizes the prediction error energy ?

$$\begin{aligned}\varepsilon_U^2 &= \mathrm{E}\{\,(S_n - a)^2\,\} = \mathrm{E}\{\,(S_n - \mathrm{E}\{\,S_n\,\} + \mathrm{E}\{\,S_n\,\} - a)^2\,\} \\ &= \mathrm{E}\{\,(S_n - \mathrm{E}\{\,S_n\,\})^2\,\} + \mathrm{E}\{\,(\mathrm{E}\{\,S_n\,\} - a)^2\,\} + 2\,\mathrm{E}\Big\{\,(S_n - \mathrm{E}\{\,S_n\,\})\,(\mathrm{E}\{\,S_n\,\} - a)\,\Big\} \\ &= \sigma_S^2 + \underbrace{(\mathrm{E}\{\,S_n\,\} - a)^2}_{\geq 0}\end{aligned}$$

➡ Prediction error energy is minimized by mean, i.e., by setting $a = \mathrm{E}\{\,S_n\,\}$

Minimization of mean-squared prediction error for given observation set $\mathcal{B}_n$

- Similar optimization problem: Minimize $\mathrm{E}\Big\{\,(S_n - f(\mathcal{B}_n))^2\,|\,\mathcal{B}_n\,\Big\}$

Optimal Predictor for Given Observation Set

- Question: What value a minimizes the prediction error energy ?

$$
\begin{aligned}
\varepsilon_U^2 &= \mathrm{E}\{ (S_n - a)^2 \} = \mathrm{E}\{ (S_n - \mathrm{E}\{ S_n \} + \mathrm{E}\{ S_n \} - a)^2 \} \\
&= \mathrm{E}\{ (S_n - \mathrm{E}\{ S_n \})^2 \} + \mathrm{E}\{ (\mathrm{E}\{ S_n \} - a)^2 \} + 2\,\mathrm{E}\Big\{ (S_n - \mathrm{E}\{ S_n \})\,(\mathrm{E}\{ S_n \} - a) \Big\} \\
&= \sigma_S^2 + \underbrace{(\mathrm{E}\{ S_n \} - a)^2}_{\geq 0}
\end{aligned}
$$

➔ Prediction error energy is minimized by mean, i.e., by setting $a = \mathrm{E}\{ S_n \}$

Minimization of mean-squared prediction error for given observation set $\mathcal{B}_n$

- Similar optimization problem: Minimize $\mathrm{E}\Big\{ (S_n - f(\mathcal{B}_n))^2 \,|\, \mathcal{B}_n \Big\}$

➔ Solution: **Optimal predictor is given by the conditional mean**

$$
\boxed{\hat{s}_n = f_{\mathrm{opt}}(\mathcal{B}_n) = \mathrm{E}\{ S_n \,|\, \mathcal{B}_n \}}
$$

Optimal Predictor for Given Observation Set

- Question: What value a minimizes the prediction error energy ?

$$\begin{aligned} \varepsilon_U^2 &= \mathrm{E}\{ (S_n - a)^2 \} = \mathrm{E}\{ (S_n - \mathrm{E}\{ S_n \} + \mathrm{E}\{ S_n \} - a)^2 \} \\ &= \mathrm{E}\{ (S_n - \mathrm{E}\{ S_n \})^2 \} + \mathrm{E}\{ (\mathrm{E}\{ S_n \} - a)^2 \} + 2\,\mathrm{E}\Big\{ (S_n - \mathrm{E}\{ S_n \}) (\mathrm{E}\{ S_n \} - a) \Big\} \\ &= \sigma_S^2 + \underbrace{(\mathrm{E}\{ S_n \} - a)^2}_{\geq 0} \end{aligned}$$

➞ Prediction error energy is minimized by mean, i.e., by setting $a = \mathrm{E}\{ S_n \}$

Minimization of mean-squared prediction error for given observation set $\mathcal{B}_n$

- Similar optimization problem: Minimize $\mathrm{E}\Big\{ (S_n - f(\mathcal{B}_n))^2 \,|\, \mathcal{B}_n \Big\}$

➞ Solution: **Optimal predictor is given by the conditional mean**

$$\boxed{\hat{s}_n = f_{\mathrm{opt}}(\mathcal{B}_n) = \mathrm{E}\{ S_n \,|\, \mathcal{B}_n \}}$$

➞ General case requires storage of large tables (often impractical and complex)

Optimal Predictor for Autoregressive Sources

Autoregressive Sources of Order m

- Good probabilistic model for many real signals: **AR(m) process**

$$S_n = \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \qquad (Z_n \text{ is zero-mean iid process})$$

Optimal Predictor for Autoregressive Sources

Autoregressive Sources of Order m

- Good probabilistic model for many real signals: **AR(m) process**

$$S_n = \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \qquad (Z_n \text{ is zero-mean iid process})$$

➡ Optimal predictor if we know the entire past, i.e., $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots\}$

$$\mathrm{E}\{\, S_n \,|\, \mathcal{B}_n \,\}$$

Optimal Predictor for Autoregressive Sources

Autoregressive Sources of Order m

- Good probabilistic model for many real signals: **AR(m) process**

$$S_n = \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \qquad (Z_n \text{ is zero-mean iid process})$$

➡ Optimal predictor if we know the entire past, i.e., $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots\}$

$$\mathrm{E}\{\, S_n \,|\, \mathcal{B}_n \,\} \;=\; \mathrm{E}\left\{ \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \;\middle|\; s_{n-1}, s_{n-2}, \cdots \right\}$$

Optimal Predictor for Autoregressive Sources

Autoregressive Sources of Order m

- Good probabilistic model for many real signals: **AR(m) process**

$$S_n = \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \qquad (Z_n \text{ is zero-mean iid process})$$

➡ Optimal predictor if we know the entire past, i.e., $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots\}$

$$\begin{aligned} \mathrm{E}\{\, S_n \,|\, \mathcal{B}_n \,\} &= \mathrm{E}\left\{ \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} \\ &= \mu_s \left(1 - \sum_{k=1}^{m} a_k \right) + \sum_{k=1}^{m} a_k \cdot \mathrm{E}\left\{ S_{n-k} \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} + \mathrm{E}\left\{ Z_n \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} \end{aligned}$$

Optimal Predictor for Autoregressive Sources

Autoregressive Sources of Order m

- Good probabilistic model for many real signals: **AR(m) process**

$$S_n = \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \qquad (Z_n \text{ is zero-mean iid process})$$

➞ Optimal predictor if we know the entire past, i.e., $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots\}$

$$\begin{aligned}
\mathrm{E}\{\, S_n \,|\, \mathcal{B}_n \,\} &= \mathrm{E}\left\{ \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} \\
&= \mu_s \left(1 - \sum_{k=1}^{m} a_k \right) + \sum_{k=1}^{m} a_k \cdot \mathrm{E}\left\{ S_{n-k} \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} + \mathrm{E}\left\{ Z_n \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} \\
&= a_0 + \sum_{k=1}^{m} a_k \cdot s_{n-k} \qquad \text{with} \quad a_0 = \mu_s \left(1 - \sum_{k=1}^{m} a_k \right)
\end{aligned}$$

Optimal Predictor for Autoregressive Sources

Autoregressive Sources of Order m

- Good probabilistic model for many real signals: **AR(m) process**

$$S_n = \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \qquad (Z_n \text{ is zero-mean iid process})$$

➡ Optimal predictor if we know the entire past, i.e., $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, \cdots\}$

$$\begin{aligned}
\mathrm{E}\{\, S_n \,|\, \mathcal{B}_n \,\} &= \mathrm{E}\left\{ \mu_s + \sum_{k=1}^{m} a_k \cdot (S_{n-k} - \mu_s) + Z_n \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} \\
&= \mu_s \left(1 - \sum_{k=1}^{m} a_k \right) + \sum_{k=1}^{m} a_k \cdot \mathrm{E}\left\{ S_{n-k} \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} + \mathrm{E}\left\{ Z_n \,\middle|\, s_{n-1}, s_{n-2}, \cdots \right\} \\
&= a_0 + \sum_{k=1}^{m} a_k \cdot s_{n-k} \qquad \text{with} \quad a_0 = \mu_s \left(1 - \sum_{k=1}^{m} a_k \right)
\end{aligned}$$

➡ **Optimal predictor is an affine function of the past m samples**

Variance and Mean for Affine Prediction

Affine Predictor: $$\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

Variance and Mean for Affine Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

- **Mean μ_U of prediction error**

$$\mu_U^2 = \mathrm{E}\{\, U_n \,\}$$

Variance and Mean for Affine Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

- **Mean μ_U of prediction error**

$$\mu_U^2 = \mathrm{E}\{\, U_n \,\} \;=\; \mathrm{E}\Big\{\, S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k \,\Big\}$$

Variance and Mean for Affine Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

- **Mean μ_U of prediction error**

$$\mu_U^2 = \mathrm{E}\{\, U_n \,\} \;=\; \mathrm{E}\Big\{\, S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k \,\Big\}$$
$$= \mu_S \left(1 - \textstyle\sum_{k=1}^{K} a_k\right) - a_0$$

Variance and Mean for Affine Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

- **Mean μ_U of prediction error**

$$\mu_U^2 = \mathrm{E}\{\, U_n \,\} \;=\; \mathrm{E}\Big\{\, S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k \,\Big\}$$
$$= \mu_S \left(1 - \textstyle\sum_{k=1}^{K} a_k\right) - a_0$$

➡ Can be forced to $\mu_U = 0$ by setting $a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k\right)$

Variance and Mean for Affine Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

- **Mean μ_U of prediction error**

$$\mu_U^2 = \mathrm{E}\{\, U_n \,\} \;=\; \mathrm{E}\Big\{\, S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k \,\Big\}$$
$$= \mu_S \left(1 - \textstyle\sum_{k=1}^{K} a_k\right) - a_0$$

➡ Can be forced to $\mu_U = 0$ by setting $a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k\right)$

- **Variance σ_U^2 of predictor error** does not depend on constant offset a_0

$$\sigma_U^2 \;=\; \mathrm{E}\{\, (U_n - \mathrm{E}\{\, U_n \,\})^2 \,\}$$

Variance and Mean for Affine Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

- **Mean μ_U of prediction error**

$$\mu_U^2 = \mathrm{E}\{\, U_n \,\} \;=\; \mathrm{E}\Big\{\, S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k \,\Big\}$$
$$= \mu_S \left(1 - \textstyle\sum_{k=1}^{K} a_k\right) - a_0$$

➡ Can be forced to $\mu_U = 0$ by setting $a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k\right)$

- **Variance σ_U^2 of predictor error** does not depend on constant offset a_0

$$\sigma_U^2 \;=\; \mathrm{E}\{\, (U_n - \mathrm{E}\{\, U_n \,\})^2 \,\} = \mathrm{E}\Big\{ \Big(S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k - \mathrm{E}\Big\{\, S_n - a_0 - \sum_{k=1}^{K} a_k\, B_k \,\Big\}\Big)^2 \Big\}$$

Variance and Mean for Affine Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

- **Mean μ_U of prediction error**

$$\mu_U^2 = \mathrm{E}\{\, U_n \,\} \;=\; \mathrm{E}\Big\{\, S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k \,\Big\}$$
$$= \mu_S \left(1 - \textstyle\sum_{k=1}^{K} a_k\right) - a_0$$

➡ Can be forced to $\mu_U = 0$ by setting $a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k\right)$

- **Variance σ_U^2 of predictor error** does not depend on constant offset a_0

$$\sigma_U^2 \;=\; \mathrm{E}\{\, (U_n - \mathrm{E}\{\, U_n \,\})^2 \,\} = \mathrm{E}\Big\{ \Big(S_n - a_0 - \textstyle\sum_{k=1}^{K} a_k\, B_k - \mathrm{E}\Big\{\, S_n - a_0 - \sum_{k=1}^{K} a_k\, B_k \,\Big\}\Big)^2 \Big\}$$
$$= \mathrm{E}\Big\{ \Big(\Big(S_n - \textstyle\sum_{k=1}^{K} a_k\, B_k \Big) - \mu_S \Big(1 - \sum_{k=1}^{K} a_k\Big)\Big)^2 \Big\} \neq f(a_0)$$

Affine and Linear Prediction

Affine Predictor: $$\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

Affine and Linear Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

➡ For a minimization of the prediction error variance σ_U^2, a linear predictor is sufficient

$$\hat{s}_n = \sum_{k=1}^{K} a_k \cdot b_k$$

Affine and Linear Prediction

Affine Predictor: $$\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

➡ For a minimization of the prediction error variance σ_U^2, a linear predictor is sufficient

$$\hat{s}_n = \sum_{k=1}^{K} a_k \cdot b_k$$

➡ The mean μ_U of the prediction error can be forced to zero by additionally setting

$$a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k \right)$$

Affine and Linear Prediction

Affine Predictor: $\boxed{\hat{s}_n = a_0 + \sum_{k=1}^{K} a_k \cdot b_k}$ for any observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

➡ For a minimization of the prediction error variance σ_U^2, a linear predictor is sufficient

$$\hat{s}_n = \sum_{k=1}^{K} a_k \cdot b_k$$

➡ The mean μ_U of the prediction error can be forced to zero by additionally setting

$$a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k \right)$$

➡ **How can we derive optimal prediction parameters $\{a_1, a_2, \cdots, a_K\}$ for a given observation set and a given source ?**

Consider: Linear Prediction and Minimization of Variance

Vector Notation (for simplifying derivations)

- Observation set: $\boldsymbol{B}_n = (B_1, B_2, \cdots, B_K)^{\mathrm{T}}$

Consider: Linear Prediction and Minimization of Variance

Vector Notation (for simplifying derivations)

- Observation set: $\boldsymbol{B}_n = (B_1, B_2, \cdots, B_K)^{\mathrm{T}}$
- Prediction parameters: $\boldsymbol{a} = (a_1, a_2, \cdots, a_K)^{\mathrm{T}}$

Consider: Linear Prediction and Minimization of Variance

Vector Notation (for simplifying derivations)

- Observation set: $\boldsymbol{B}_n = (B_1, B_2, \cdots, B_K)^{\mathrm{T}}$
- Prediction parameters: $\boldsymbol{a} = (a_1, a_2, \cdots, a_K)^{\mathrm{T}}$

➡ Linear predictor: $\hat{S}_n = \boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{B}_n$

Consider: Linear Prediction and Minimization of Variance

Vector Notation (for simplifying derivations)

- Observation set: $\boldsymbol{B}_n = (B_1, B_2, \cdots, B_K)^{\mathrm{T}}$
- Prediction parameters: $\boldsymbol{a} = (a_1, a_2, \cdots, a_K)^{\mathrm{T}}$

➡ Linear predictor: $\hat{S}_n = \boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{B}_n$

➡ Predictor error: $U_n = S_n - \hat{S}_n = S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n$

Consider: Linear Prediction and Minimization of Variance

Vector Notation (for simplifying derivations)

- Observation set: $\boldsymbol{B}_n = (B_1, B_2, \cdots, B_K)^{\mathrm{T}}$
- Prediction parameters: $\boldsymbol{a} = (a_1, a_2, \cdots, a_K)^{\mathrm{T}}$

➡ Linear predictor: $\hat{S}_n = \boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{B}_n$

➡ Predictor error: $U_n = S_n - \hat{S}_n = S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n$

Optimization Problem

- Minimization of prediction error variance

$$\sigma_U^2(\boldsymbol{a}) = \mathrm{E}\Big\{ \big(U_n - \mathrm{E}\{ U_n \} \big)^2 \Big\}$$

Consider: Linear Prediction and Minimization of Variance

Vector Notation (for simplifying derivations)

- Observation set: $\boldsymbol{B}_n = (B_1, B_2, \cdots, B_K)^{\mathrm{T}}$
- Prediction parameters: $\boldsymbol{a} = (a_1, a_2, \cdots, a_K)^{\mathrm{T}}$

➡ Linear predictor: $\hat{S}_n = \boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{B}_n$

➡ Predictor error: $U_n = S_n - \hat{S}_n = S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n$

Optimization Problem

- Minimization of prediction error variance

$$\sigma_U^2(\boldsymbol{a}) = \mathrm{E}\left\{ \Big(U_n - \mathrm{E}\{\, U_n \,\} \Big)^2 \right\} = \mathrm{E}\left\{ \Big(S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n - \mathrm{E}\{\, S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n \,\} \Big)^2 \right\}$$

Consider: Linear Prediction and Minimization of Variance

Vector Notation (for simplifying derivations)

- Observation set: $\boldsymbol{B}_n = (B_1, B_2, \cdots, B_K)^{\mathrm{T}}$
- Prediction parameters: $\boldsymbol{a} = (a_1, a_2, \cdots, a_K)^{\mathrm{T}}$

➡ Linear predictor: $\hat{S}_n = \boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{B}_n$

➡ Predictor error: $U_n = S_n - \hat{S}_n = S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n$

Optimization Problem

- Minimization of prediction error variance

$$
\begin{aligned}
\sigma_U^2(\boldsymbol{a}) &= \mathrm{E}\Big\{ \big(U_n - \mathrm{E}\{ U_n \} \big)^2 \Big\} = \mathrm{E}\Big\{ \big(S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n - \mathrm{E}\{ S_n - \boldsymbol{a}^{\mathrm{T}} \boldsymbol{B}_n \} \big)^2 \Big\} \\
&= \mathrm{E}\Big\{ \Big(\big(S_n - \mathrm{E}\{ S_n \} \big) - \boldsymbol{a}^{\mathrm{T}} \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big) \Big)^2 \Big\}
\end{aligned}
$$

Reformulate Prediction Error Variance

- Prediction error variance

$$\sigma_U^2(\boldsymbol{a}) \;=\; \mathrm{E}\left\{ \left(\left(S_n - \mathrm{E}\{\, S_n \,\} \right) - \boldsymbol{a}^{\mathrm{T}} \left(\boldsymbol{B}_n - \mathrm{E}\{\, \boldsymbol{B}_n \,\} \right) \right)^2 \right\}$$

Reformulate Prediction Error Variance

- Prediction error variance

$$
\begin{aligned}
\sigma_U^2(\boldsymbol{a}) &= \mathrm{E}\left\{ \left(\left(S_n - \mathrm{E}\{\, S_n \,\} \right) - \boldsymbol{a}^{\mathrm{T}} \left(\boldsymbol{B}_n - \mathrm{E}\{\, \boldsymbol{B}_n \,\} \right) \right)^2 \right\} \\
&= \mathrm{E}\left\{ \left(S_n - \mathrm{E}\{\, S_n \,\} \right)^2 \right\} - 2\, \boldsymbol{a}^{\mathrm{T}} \cdot \mathrm{E}\left\{ \left(S_n - \mathrm{E}\{\, S_n \,\} \right) \left(\boldsymbol{B}_n - \mathrm{E}\{\, \boldsymbol{B}_n \,\} \right) \right\} \\
&\quad + \boldsymbol{a}^{\mathrm{T}} \cdot \mathrm{E}\left\{ \left(\boldsymbol{B}_n - \mathrm{E}\{\, \boldsymbol{B}_n \,\} \right) \left(\boldsymbol{B}_n - \mathrm{E}\{\, \boldsymbol{B}_n \,\} \right)^{\mathrm{T}} \right\} \cdot \boldsymbol{a}
\end{aligned}
$$

Reformulate Prediction Error Variance

- Prediction error variance

$$\begin{aligned}
\sigma_U^2(\boldsymbol{a}) &= \mathrm{E}\Big\{ \Big(\big(S_n - \mathrm{E}\{ S_n \} \big) - \boldsymbol{a}^{\mathrm{T}} \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big) \Big)^2 \Big\} \\
&= \mathrm{E}\Big\{ \big(S_n - \mathrm{E}\{ S_n \} \big)^2 \Big\} - 2\, \boldsymbol{a}^{\mathrm{T}} \cdot \mathrm{E}\Big\{ \big(S_n - \mathrm{E}\{ S_n \} \big) \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big) \Big\} \\
&\quad + \boldsymbol{a}^{\mathrm{T}} \cdot \mathrm{E}\Big\{ \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big) \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big)^{\mathrm{T}} \Big\} \cdot \boldsymbol{a} \\
&= \sigma_S^2 - 2\, \boldsymbol{a}^{\mathrm{T}}\, \boldsymbol{c} + \boldsymbol{a}^{\mathrm{T}}\, \boldsymbol{C}_B\, \boldsymbol{a}
\end{aligned}$$

with $\boldsymbol{C}_B$ and $\boldsymbol{c}$ being given by

$$\begin{aligned}
\boldsymbol{C}_B &= \mathrm{E}\Big\{ \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big) \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big)^{\mathrm{T}} \Big\} \\
\boldsymbol{c} &= \mathrm{E}\Big\{ \big(S_n - \mathrm{E}\{ S_n \} \big) \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big) \Big\}
\end{aligned}$$

Auto-Covariance Matrix of the Observation Set

$$\boldsymbol{C}_B = \mathrm{E}\Big\{ \big(\boldsymbol{B}_n - \mathrm{E}\{\,\boldsymbol{B}_n\,\}\big)\big(\boldsymbol{B}_n - \mathrm{E}\{\,\boldsymbol{B}_n\,\}\big)^{\mathrm{T}} \Big\} = \sigma_S^2 \cdot \begin{bmatrix} 1 & \varrho_{1,2} & \varrho_{1,3} & \cdots & \varrho_{1,K} \\ \varrho_{2,1} & 1 & \varrho_{2,3} & \cdots & \varrho_{2,K} \\ \varrho_{3,1} & \varrho_{3,2} & 1 & \cdots & \varrho_{3,K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \varrho_{K,1} & \varrho_{K,2} & \varrho_{K,3} & \cdots & 1 \end{bmatrix}$$

- Each entry represents the correlation coefficient between two samples of the observation set

$$\varrho_{a,b} = \varrho_{b,a} = \frac{\mathrm{cov}(B_a, B_b)}{\sigma_S^2} = \frac{\mathrm{E}\Big\{ \big(B_a - \mu_S\big)\big(B_b - \mu_S\big) \Big\}}{\mathrm{E}\Big\{ \big(S_n - \mu_S\big)^2 \Big\}}$$

Auto-Covariance Matrix of the Observation Set

$$\boldsymbol{C}_B = \mathrm{E}\Big\{ \big(\boldsymbol{B}_n - \mathrm{E}\{\,\boldsymbol{B}_n\,\}\big)\big(\boldsymbol{B}_n - \mathrm{E}\{\,\boldsymbol{B}_n\,\}\big)^{\mathrm{T}} \Big\} = \sigma_S^2 \cdot \begin{bmatrix} 1 & \varrho_{1,2} & \varrho_{1,3} & \cdots & \varrho_{1,K} \\ \varrho_{2,1} & 1 & \varrho_{2,3} & \cdots & \varrho_{2,K} \\ \varrho_{3,1} & \varrho_{3,2} & 1 & \cdots & \varrho_{3,K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \varrho_{K,1} & \varrho_{K,2} & \varrho_{K,3} & \cdots & 1 \end{bmatrix}$$

- Each entry represents the correlation coefficient between two samples of the observation set

$$\varrho_{a,b} = \varrho_{b,a} = \frac{\mathrm{cov}(B_a, B_b)}{\sigma_S^2} = \frac{\mathrm{E}\Big\{ \big(B_a - \mu_S\big)\big(B_b - \mu_S\big) \Big\}}{\mathrm{E}\Big\{ \big(S_n - \mu_S\big)^2 \Big\}}$$

➡ Property of the source (and choice of observation set)

Auto-Covariance Matrix of the Observation Set

$$\boldsymbol{C}_B = \mathrm{E}\Big\{ \big(\boldsymbol{B}_n - \mathrm{E}\{\,\boldsymbol{B}_n\,\}\big)\big(\boldsymbol{B}_n - \mathrm{E}\{\,\boldsymbol{B}_n\,\}\big)^{\mathrm{T}} \Big\} = \sigma_S^2 \cdot \begin{bmatrix} 1 & \varrho_{1,2} & \varrho_{1,3} & \cdots & \varrho_{1,K} \\ \varrho_{2,1} & 1 & \varrho_{2,3} & \cdots & \varrho_{2,K} \\ \varrho_{3,1} & \varrho_{3,2} & 1 & \cdots & \varrho_{3,K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \varrho_{K,1} & \varrho_{K,2} & \varrho_{K,3} & \cdots & 1 \end{bmatrix}$$

- Each entry represents the correlation coefficient between two samples of the observation set

$$\varrho_{a,b} = \varrho_{b,a} = \frac{\mathrm{cov}(B_a, B_b)}{\sigma_S^2} = \frac{\mathrm{E}\Big\{ \big(B_a - \mu_S\big)\big(B_b - \mu_S\big) \Big\}}{\mathrm{E}\Big\{ \big(S_n - \mu_S\big)^2 \Big\}}$$

➡ Property of the source (and choice of observation set)

➡ Can be measured for given signal (or set of signals)

Cross-Covariance Vector of the Observation Set and Current Sample

$$\boldsymbol{c} = \mathrm{E}\Big\{ \big(S_n - \mathrm{E}\{ S_n \}\big)\big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \}\big) \Big\} = \sigma_S^2 \cdot \begin{bmatrix} \varrho_{c,1} \\ \varrho_{c,2} \\ \varrho_{c,3} \\ \vdots \\ \varrho_{c,K} \end{bmatrix}$$

- Each entry represents the correlation coefficient between the sample to be predicted and a sample of the observation set

$$\varrho_{c,k} = \frac{\mathrm{cov}(S_n, B_k)}{\sigma_S^2} = \frac{\mathrm{E}\Big\{ \big(S_n - \mu_S\big)\big(B_k - \mu_S\big) \Big\}}{\mathrm{E}\Big\{ \big(S_n - \mu_S\big)^2 \Big\}}$$

Cross-Covariance Vector of the Observation Set and Current Sample

$$\boldsymbol{c} = \mathrm{E}\Big\{ \big(S_n - \mathrm{E}\{ S_n \} \big) \big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \big) \Big\} = \sigma_S^2 \cdot \begin{bmatrix} \varrho_{c,1} \\ \varrho_{c,2} \\ \varrho_{c,3} \\ \vdots \\ \varrho_{c,K} \end{bmatrix}$$

- Each entry represents the correlation coefficient between the sample to be predicted and a sample of the observation set

$$\varrho_{c,k} = \frac{\mathrm{cov}(S_n, B_k)}{\sigma_S^2} = \frac{\mathrm{E}\Big\{ \big(S_n - \mu_S \big) \big(B_k - \mu_S \big) \Big\}}{\mathrm{E}\Big\{ \big(S_n - \mu_S \big)^2 \Big\}}$$

➡ Property of the source (and choice of observation set)

Cross-Covariance Vector of the Observation Set and Current Sample

$$\boldsymbol{c} = \mathrm{E}\Big\{ \Big(S_n - \mathrm{E}\{ S_n \} \Big) \Big(\boldsymbol{B}_n - \mathrm{E}\{ \boldsymbol{B}_n \} \Big) \Big\} = \sigma_S^2 \cdot \begin{bmatrix} \varrho_{c,1} \\ \varrho_{c,2} \\ \varrho_{c,3} \\ \vdots \\ \varrho_{c,K} \end{bmatrix}$$

- Each entry represents the correlation coefficient between the sample to be predicted and a sample of the observation set

$$\varrho_{c,k} = \frac{\mathrm{cov}(S_n, B_k)}{\sigma_S^2} = \frac{\mathrm{E}\Big\{ \Big(S_n - \mu_S \Big) \Big(B_k - \mu_S \Big) \Big\}}{\mathrm{E}\Big\{ \Big(S_n - \mu_S \Big)^2 \Big\}}$$

➡ Property of the source (and choice of observation set)

➡ Can be measured for given signal (or set of signals)

Optimal Linear and Affine Prediction

- Goal: Minimization of prediction error variance σ_U^2

$$\sigma_U^2(\boldsymbol{a}) = \sigma_S^2 - 2\,\boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{c} + \boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{C}_B\,\boldsymbol{a}$$

Optimal Linear and Affine Prediction

- Goal: Minimization of prediction error variance σ_U^2

$$\sigma_U^2(\boldsymbol{a}) = \sigma_S^2 - 2\,\boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{c} + \boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{C}_B\,\boldsymbol{a}$$

- Set derivative with respect to $\boldsymbol{a}$ equal to zero

$$\frac{\partial}{\partial\,\boldsymbol{a}}\,\sigma_U^2(\boldsymbol{a}) = -2\,\boldsymbol{c} + 2\,\boldsymbol{C}_B \cdot \boldsymbol{a} = 0$$

Optimal Linear and Affine Prediction

- Goal: Minimization of prediction error variance σ_U^2

$$\sigma_U^2(\boldsymbol{a}) = \sigma_S^2 - 2\,\boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{c} + \boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{C}_B\,\boldsymbol{a}$$

- Set derivative with respect to $\boldsymbol{a}$ equal to zero

$$\frac{\partial}{\partial\,\boldsymbol{a}}\,\sigma_U^2(\boldsymbol{a}) = -2\,\boldsymbol{c} + 2\,\boldsymbol{C}_B \cdot \boldsymbol{a} = 0$$

➡ **Yule-Walker equations** (linear equation system)

optimal predictor $\boldsymbol{a}$ solves $\boxed{\boldsymbol{C}_B \cdot \boldsymbol{a} = \boldsymbol{c}}$

Optimal Linear and Affine Prediction

- Goal: Minimization of prediction error variance σ_U^2

$$\sigma_U^2(\boldsymbol{a}) = \sigma_S^2 - 2\,\boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{c} + \boldsymbol{a}^{\mathrm{T}}\,\boldsymbol{C}_B\,\boldsymbol{a}$$

- Set derivative with respect to $\boldsymbol{a}$ equal to zero

$$\frac{\partial}{\partial\,\boldsymbol{a}}\,\sigma_U^2(\boldsymbol{a}) = -2\,\boldsymbol{c} + 2\,\boldsymbol{C}_B \cdot \boldsymbol{a} = 0$$

➡ **Yule-Walker equations** (linear equation system)

optimal predictor $\boldsymbol{a}$ solves $\boxed{\boldsymbol{C}_B \cdot \boldsymbol{a} = \boldsymbol{c}}$

➡ Optimal affine prediction: Additionally, set

$$a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k \right)$$

Summary: Derivation of Optimal Affine Predictor for Given Source

1. Choose suitable observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$

Summary: Derivation of Optimal Affine Predictor for Given Source

1. Choose suitable observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$
2. Determine mean μ_S of sources and required correlation coefficients $\varrho_{k,\ell}$

Summary: Derivation of Optimal Affine Predictor for Given Source

1. Choose suitable observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$
2. Determine mean μ_S of sources and required correlation coefficients $\varrho_{k,\ell}$
3. Solve linear equation system for determining prediction parameters $a_1, a_2, \cdots, a_K$

$$\begin{bmatrix} 1 & \varrho_{1,2} & \varrho_{1,3} & \cdots & \varrho_{1,K} \\ \varrho_{2,1} & 1 & \varrho_{2,3} & \cdots & \varrho_{2,K} \\ \varrho_{3,1} & \varrho_{3,2} & 1 & \cdots & \varrho_{3,K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \varrho_{K,1} & \varrho_{K,2} & \varrho_{K,3} & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_K \end{bmatrix} = \begin{bmatrix} \varrho_{c,1} \\ \varrho_{c,2} \\ \varrho_{c,3} \\ \vdots \\ \varrho_{c,K} \end{bmatrix}$$

Summary: Derivation of Optimal Affine Predictor for Given Source

1. Choose suitable observation set $\mathcal{B}_n = \{b_1, b_2, \cdots, b_K\}$
2. Determine mean μ_S of sources and required correlation coefficients $\varrho_{k,\ell}$
3. Solve linear equation system for determining prediction parameters $a_1, a_2, \cdots, a_K$

$$\begin{bmatrix} 1 & \varrho_{1,2} & \varrho_{1,3} & \cdots & \varrho_{1,K} \\ \varrho_{2,1} & 1 & \varrho_{2,3} & \cdots & \varrho_{2,K} \\ \varrho_{3,1} & \varrho_{3,2} & 1 & \cdots & \varrho_{3,K} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \varrho_{K,1} & \varrho_{K,2} & \varrho_{K,3} & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_K \end{bmatrix} = \begin{bmatrix} \varrho_{c,1} \\ \varrho_{c,2} \\ \varrho_{c,3} \\ \vdots \\ \varrho_{c,K} \end{bmatrix}$$

4. Determine constant offset

$$a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k \right)$$

Example: Prediction of Audio Signals using Three Preceding Samples

1. Choice of observation set for sample s_n: $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, s_{n-3}\}$

Example: Prediction of Audio Signals using Three Preceding Samples

1. Choice of observation set for sample s_n: $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, s_{n-3}\}$
2. Need to determine mean μ_S and three correlation coefficients ϱ_k (for $k = 1, 2, 3$)

$$\mu_S = \mathrm{E}\{ S_n \}$$

$$\varrho_k = \frac{\mathrm{E}\{ (S_n - \mu_S)(S_{n-k} - \mu_S) \}}{\mathrm{E}\{ (S_n - \mu_S)^2 \}}$$

Example: Prediction of Audio Signals using Three Preceding Samples

1. Choice of observation set for sample s_n: $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, s_{n-3}\}$
2. Need to determine mean μ_S and three correlation coefficients ϱ_k (for $k = 1, 2, 3$)

$$\mu_S = \mathrm{E}\{ S_n \} = \frac{1}{N} \sum_{n=1}^{N} s_n$$

$$\varrho_k = \frac{\mathrm{E}\{ (S_n - \mu_S)(S_{n-k} - \mu_S) \}}{\mathrm{E}\{ (S_n - \mu_S)^2 \}}$$

Example: Prediction of Audio Signals using Three Preceding Samples

1. Choice of observation set for sample s_n: $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, s_{n-3}\}$
2. Need to determine mean μ_S and three correlation coefficients ϱ_k (for $k = 1, 2, 3$)

$$\mu_S = \mathrm{E}\{\, S_n \,\} = \frac{1}{N} \sum_{n=1}^{N} s_n$$

$$\varrho_k = \frac{\mathrm{E}\{\, (S_n - \mu_S)(S_{n-k} - \mu_S) \,\}}{\mathrm{E}\{\, (S_n - \mu_S)^2 \,\}} = \frac{\sum_{n=k}^{N} (s_n - \mu_S)(s_{n-k} - \mu_S)}{\sum_{n=k}^{N} (s_n - \mu_S)^2}$$

Example: Prediction of Audio Signals using Three Preceding Samples

1. Choice of observation set for sample s_n: $\mathcal{B}_n = \{s_{n-1}, s_{n-2}, s_{n-3}\}$
2. Need to determine mean μ_S and three correlation coefficients ϱ_k (for $k = 1, 2, 3$)

$$\mu_S = \mathrm{E}\{\, S_n \,\} = \frac{1}{N} \sum_{n=1}^{N} s_n$$

$$\varrho_k = \frac{\mathrm{E}\{\, (S_n - \mu_S)(S_{n-k} - \mu_S) \,\}}{\mathrm{E}\{\, (S_n - \mu_S)^2 \,\}} = \frac{\sum_{n=k}^{N} (s_n - \mu_S)(s_{n-k} - \mu_S)}{\sum_{n=k}^{N} (s_n - \mu_S)^2}$$

Example

$$\mu_S = -37.6917$$
$$\varrho_1 = 0.9581$$
$$\varrho_2 = 0.8619$$
$$\varrho_3 = 0.7564$$

Example: Prediction of Audio Signals using Three Preceding Samples

3 Solve linear equation system

$$\begin{bmatrix} 1 & \varrho_1 & \varrho_2 \\ \varrho_1 & 1 & \varrho_1 \\ \varrho_2 & \varrho_1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \varrho_1 \\ \varrho_2 \\ \varrho_3 \end{bmatrix} \qquad \Longrightarrow \qquad \begin{aligned} a_1 &= 1.9409 \\ a_2 &= -1.4580 \\ a_3 &= 0.4804 \end{aligned}$$

Example: Prediction of Audio Signals using Three Preceding Samples

3 Solve linear equation system

$$\begin{bmatrix} 1 & \varrho_1 & \varrho_2 \\ \varrho_1 & 1 & \varrho_1 \\ \varrho_2 & \varrho_1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \varrho_1 \\ \varrho_2 \\ \varrho_3 \end{bmatrix} \qquad \Longrightarrow \qquad \begin{aligned} a_1 &= 1.9409 \\ a_2 &= -1.4580 \\ a_3 &= 0.4804 \end{aligned}$$

4 Determine constant offset

$$a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k \right) \qquad \Longrightarrow \qquad a_0 = -1.3833$$

Example: Prediction of Audio Signals using Three Preceding Samples

3 Solve linear equation system

$$\begin{bmatrix} 1 & \varrho_1 & \varrho_2 \\ \varrho_1 & 1 & \varrho_1 \\ \varrho_2 & \varrho_1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \varrho_1 \\ \varrho_2 \\ \varrho_3 \end{bmatrix} \quad \Longrightarrow \quad \begin{aligned} a_1 &= 1.9409 \\ a_2 &= -1.4580 \\ a_3 &= 0.4804 \end{aligned}$$

4 Determine constant offset

$$a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k \right) \quad \Longrightarrow \quad a_0 = -1.3833$$

➡ Predictor is given by

$$\begin{aligned} \hat{s}_n &= \mathrm{round}\Big(a_0 + a_1 \cdot s_{n-1} + a_2 \cdot s_{n-2} + a_3 \cdot s_{n-3} \Big) \\ &= \mathrm{round}\Big(-1.3833 + 1.9409 \cdot s_{n-1} - 1.4580 \cdot s_{n-2} + 0.4804 \cdot s_{n-3} \Big) \end{aligned}$$

Example: Prediction of Audio Signals using Three Preceding Samples

3 Solve linear equation system

$$\begin{bmatrix} 1 & \varrho_1 & \varrho_2 \\ \varrho_1 & 1 & \varrho_1 \\ \varrho_2 & \varrho_1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \varrho_1 \\ \varrho_2 \\ \varrho_3 \end{bmatrix} \qquad \Longrightarrow \qquad \begin{aligned} a_1 &= 1.9409 \\ a_2 &= -1.4580 \\ a_3 &= 0.4804 \end{aligned}$$

4 Determine constant offset

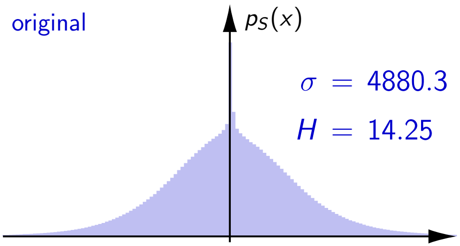
$$a_0 = \mu_S \left(1 - \sum_{k=1}^{K} a_k \right) \qquad \Longrightarrow \qquad a_0 = -1.3833$$

➡ **Predictor is given by**

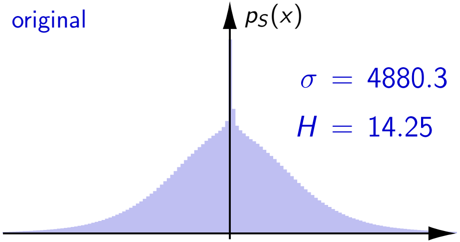
$$\begin{aligned} \hat{s}_n &= \mathrm{round}\big(a_0 + a_1 \cdot s_{n-1} + a_2 \cdot s_{n-2} + a_3 \cdot s_{n-3} \big) \\ &= \mathrm{round}\big(-1.3833 + 1.9409 \cdot s_{n-1} - 1.4580 \cdot s_{n-2} + 0.4804 \cdot s_{n-3} \big) \end{aligned}$$

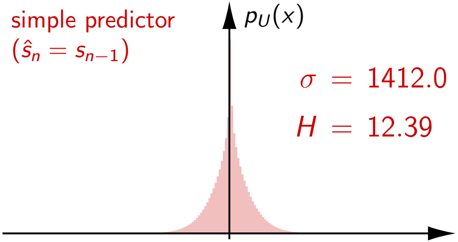
Note: Predictor must be rounded to an integer (want to apply lossless coding)

Example: Prediction of Audio Signals using Three Preceding Samples



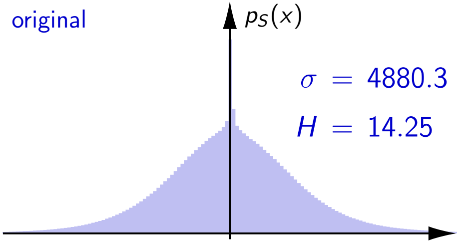
Example: Prediction of Audio Signals using Three Preceding Samples



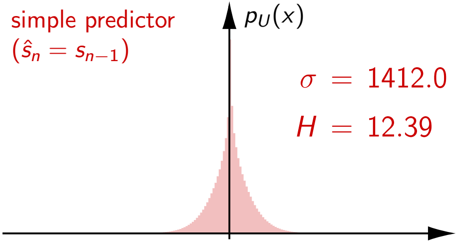


Example: Prediction of Audio Signals using Three Preceding Samples

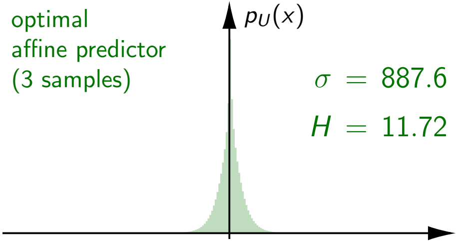
original



simple predictor
($\hat{s}_n = s_{n-1}$)



optimal
affine predictor
(3 samples)



Potential Improvements for Audio Coding

1. **Use more samples as observation set**
   - ➡ May improve the affine predictor
   - ➡ Requires transmission of more prediction parameters
   - ➡ Audio compression format **FLAC** uses up to 32 preceding samples

Potential Improvements for Audio Coding

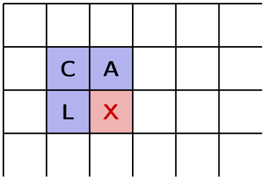
1. **Use more samples as observation set**
   - ➡ May improve the affine predictor
   - ➡ Requires transmission of more prediction parameters
   - ➡ Audio compression format **FLAC** uses up to 32 preceding samples

2. **Adapt predictor during coding**
   - Audio signals have instationary properties
   - ➡ A single predictor may not be suitable for all parts of an audio stream
   - ➡ Split data into chunks and determine best predictor for each chunk

Example: Prediction of Image Signals using Three Neighboring Samples

1 Choice of prediction structure

$$\hat{X} = a_0 + a_L \cdot L + a_A \cdot A + a_C \cdot C$$



Example: Prediction of Image Signals using Three Neighboring Samples

1 Choice of prediction structure

$$\hat{X} = a_0 + a_L \cdot L + a_A \cdot A + a_C \cdot C$$

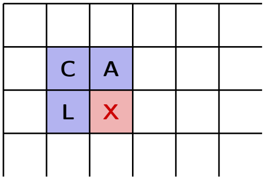
2 Determine mean and required correlation factors

$$\varrho_{\text{hor}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x-1,y]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$

$$\varrho_{\text{ver}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x,y-1]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$

$$\varrho_{\text{al}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x-1,y-1]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$

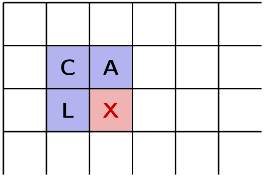
$$\varrho_{\text{ar}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x+1,y-1]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$



Example: Prediction of Image Signals using Three Neighboring Samples

1 Choice of prediction structure

$$\hat{X} = a_0 + a_L \cdot L + a_A \cdot A + a_C \cdot C$$



2 Determine mean and required correlation factors

$$\varrho_{\mathrm{hor}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x-1,y]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$

$$\varrho_{\mathrm{ver}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x,y-1]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$

$$\varrho_{\mathrm{al}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x-1,y-1]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$

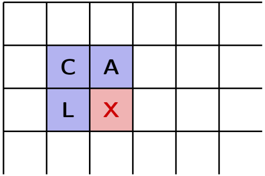
$$\varrho_{\mathrm{ar}} = \frac{\sum_{x,y}(s[x,y]-\mu_S)(s[x+1,y-1]-\mu_S)}{\sum_{x,y}(s[x,y]-\mu_S)^2}$$

$$\begin{aligned} \mu_S &= 124.05 \\ \varrho_{\mathrm{hor}} &= 0.9722 \\ \varrho_{\mathrm{ver}} &= 0.9850 \\ \varrho_{\mathrm{al}} &= 0.9598 \\ \varrho_{\mathrm{ar}} &= 0.9689 \end{aligned}$$

Example: Prediction of Image Signals using Three Neighboring Samples

3 Solve linear equation system

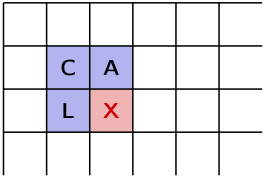
$$\begin{bmatrix} 1 & \varrho_{\mathrm{ar}} & \varrho_{\mathrm{ver}} \\ \varrho_{\mathrm{ar}} & 1 & \varrho_{\mathrm{hor}} \\ \varrho_{\mathrm{ver}} & \varrho_{\mathrm{hor}} & 1 \end{bmatrix} \cdot \begin{bmatrix} a_L \\ a_A \\ a_C \end{bmatrix} = \begin{bmatrix} \varrho_{\mathrm{hor}} \\ \varrho_{\mathrm{ver}} \\ \varrho_{\mathrm{al}} \end{bmatrix} \quad \Longrightarrow$$



Example: Prediction of Image Signals using Three Neighboring Samples

3 Solve linear equation system

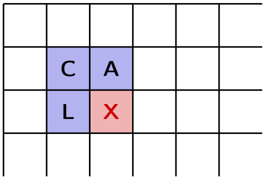
$$\begin{bmatrix} 1 & \varrho_{\mathrm{ar}} & \varrho_{\mathrm{ver}} \\ \varrho_{\mathrm{ar}} & 1 & \varrho_{\mathrm{hor}} \\ \varrho_{\mathrm{ver}} & \varrho_{\mathrm{hor}} & 1 \end{bmatrix} \cdot \begin{bmatrix} a_L \\ a_A \\ a_C \end{bmatrix} = \begin{bmatrix} \varrho_{\mathrm{hor}} \\ \varrho_{\mathrm{ver}} \\ \varrho_{\mathrm{al}} \end{bmatrix} \quad \Longrightarrow \quad \begin{aligned} a_L &= 0.5892 \\ a_A &= 0.8255 \\ a_C &= -0.4262 \end{aligned}$$



Example: Prediction of Image Signals using Three Neighboring Samples

3 Solve linear equation system

$$\begin{bmatrix} 1 & \varrho_{\text{ar}} & \varrho_{\text{ver}} \\ \varrho_{\text{ar}} & 1 & \varrho_{\text{hor}} \\ \varrho_{\text{ver}} & \varrho_{\text{hor}} & 1 \end{bmatrix} \cdot \begin{bmatrix} a_L \\ a_A \\ a_C \end{bmatrix} = \begin{bmatrix} \varrho_{\text{hor}} \\ \varrho_{\text{ver}} \\ \varrho_{\text{al}} \end{bmatrix} \quad \Longrightarrow \quad \begin{aligned} a_L &= 0.5892 \\ a_A &= 0.8255 \\ a_C &= -0.4262 \end{aligned}$$



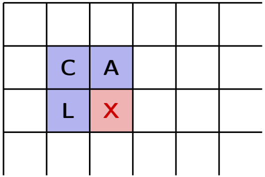
4 Determine constant offset

$$a_0 = \mu_S \left(1 - a_L - a_A - a_C\right) \quad \Longrightarrow \quad a_0 = 1.4198$$

Example: Prediction of Image Signals using Three Neighboring Samples

3 Solve linear equation system

$$\begin{bmatrix} 1 & \varrho_{\text{ar}} & \varrho_{\text{ver}} \\ \varrho_{\text{ar}} & 1 & \varrho_{\text{hor}} \\ \varrho_{\text{ver}} & \varrho_{\text{hor}} & 1 \end{bmatrix} \cdot \begin{bmatrix} a_L \\ a_A \\ a_C \end{bmatrix} = \begin{bmatrix} \varrho_{\text{hor}} \\ \varrho_{\text{ver}} \\ \varrho_{\text{al}} \end{bmatrix} \quad \Longrightarrow \quad \begin{aligned} a_L &= 0.5892 \\ a_A &= 0.8255 \\ a_C &= -0.4262 \end{aligned}$$



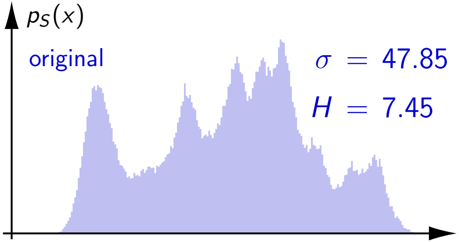
4 Determine constant offset

$$a_0 = \mu_S \left(1 - a_L - a_A - a_C\right) \quad \Longrightarrow \quad a_0 = 1.4198$$

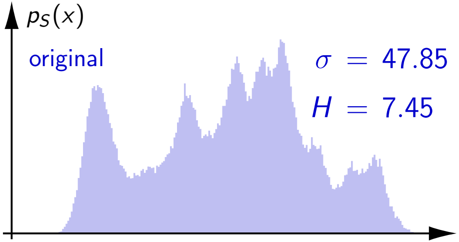
➡ Predictor is given by

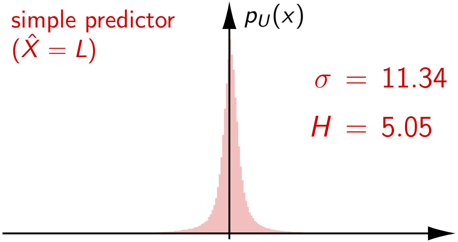
$$\begin{aligned} \hat{X} &= \operatorname{round}\Big(a_0 + a_L \cdot L + a_A \cdot A + a_C \cdot C \Big) \\ &= \operatorname{round}\Big(1.4198 + 0.5892 \cdot L + 0.8255 \cdot A - 0.4262 \cdot C \Big) \end{aligned}$$

Example: Prediction of Image Signals using Three Neighboring Samples

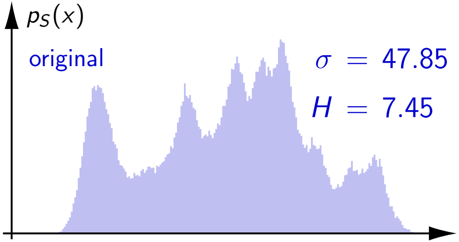


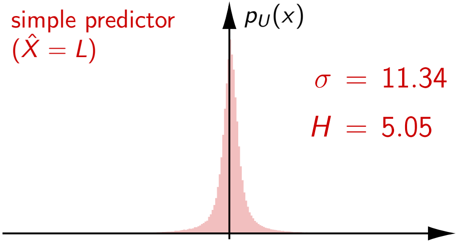
Example: Prediction of Image Signals using Three Neighboring Samples

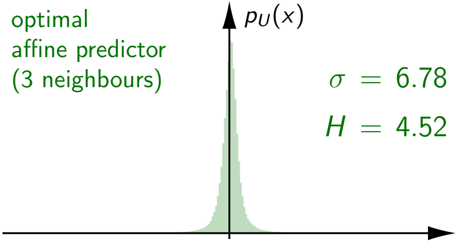




Example: Prediction of Image Signals using Three Neighboring Samples







Potential Improvements for Image Coding

1. **Use more samples as observation set**
   - ➡ May improve the affine predictor
   - ➡ Requires transmission of more prediction parameters

Potential Improvements for Image Coding

1 Use more samples as observation set

➡ May improve the affine predictor

➡ Requires transmission of more prediction parameters

2 Adapt predictor during coding

- Image signals have instationary properties: Direction of edges plays an important role

➡ Split image into blocks and choose predictor for each block

➡ May be sufficient to choose between pre-defined predictors

  - Horizontal predictor for parts with horizontal edges
  - Vertical predictor for parts with vertical edges
  - Some further predictors

Potential Improvements for Image Coding

1 Use more samples as observation set

- ➡ May improve the affine predictor
- ➡ Requires transmission of more prediction parameters

2 Adapt predictor during coding

- Image signals have instationary properties: Direction of edges plays an important role
- ➡ Split image into blocks and choose predictor for each block
- ➡ May be sufficient to choose between pre-defined predictors
  - Horizontal predictor for parts with horizontal edges
  - Vertical predictor for parts with vertical edges
  - Some further predictors

3 Non-linear Predictors

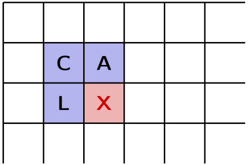
- ➡ Non-linear predictors may be able to deal with different edge directions

Examples for Non-Linear Predictors

LOCO Predictor used in JPEG-LS

- Each sample X is predicted according to

$$\hat{X} = \begin{cases} \min(L, A) & : \; C \geq \max(L, A) \\ \max(L, A) & : \; C \leq \min(L, A) \\ L + A - C & : \; \text{otherwise} \end{cases}$$

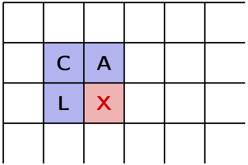


Examples for Non-Linear Predictors

LOCO Predictor used in JPEG-LS

- Each sample X is predicted according to

$$\hat{X} = \begin{cases} \min(L, A) & : \; C \geq \max(L, A) \\ \max(L, A) & : \; C \leq \min(L, A) \\ L + A - C & : \; \text{otherwise} \end{cases}$$

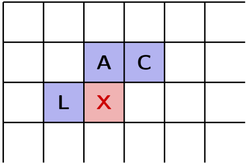


Motion Vector Prediction in Video Coding Standards

- Motion vector $\boldsymbol{m}$ is predicted by component-wise median

$$\hat{m}_x(X) \; = \; \mathrm{median}\Big(m_x(A), \; m_x(B), \; m_x(C) \Big)$$

$$\hat{m}_y(X) \; = \; \mathrm{median}\Big(m_y(A), \; m_y(B), \; m_y(C) \Big)$$



Summary of Lecture

Predictive Lossless Coding

- Entropy coding of prediction error signals $u_n = s_n - \hat{s}_n$
- Simple and effective way to exploit dependencies between neighbouring samples
- Complexity reduction relative to more general conditional entropy coding

Optimal Prediction

- Given by conditional mean for an observation set
- Complex due to requirement of large tables (similar to conditional entropy coding)

Affine and Linear Prediction

- Simple structure of predictor, low-complex implementations possible
- Optimal prediction parameters are given by solution of Yule-Walker equations
- For instationary sources (such as audio, image, video signals):
  - Determine predictor for smaller sets of samples (still large enough)
  - Determine optimal predictor or choose between set of predefined predictors

Exercise: Lossless Image Compression Challenge (Part I)

Implement an encoder and decoder for lossless coding of 8-bit color images:

1. We use the PPM format as raw data format:
    - The encoder should read the original images in PPM format.
    - The decoder should write the reconstructed images in PPM format.

   Example images (24 PPM images of the Kodak set) are provided on the course web-site (and in the KVV).

2. Use coding techniques that you learned for efficiently compressing the 8-bit color images.
    - A combination of prediction and entropy coding of the prediction errors is suggested.
    - Start with a simple (but working) approach and try to improve your codec step by step.

structure of "ppm" files:

```
P6                 // ascii (fixed)
width height       // ascii
255                // ascii (max. value)
<binary data>      // binary
```

binary data:

- pixels in raster-scan order (line by line)
- each pixel consists of three 8-bit values
    - ➡ R: red component (0..255)
    - ➡ G: green component (0..255)
    - ➡ B: blue component (0..255)
- the values R, G, B for a pixel follow each other (before the values for the next pixel)

suggestion:

- Store the red, green, and blue components of an image into separate arrays
- Code the color components independently