ERDC/GSL TR-21-13

Geotechnical and Structures Laboratory

US Army Corps
of Engineers®
Engineer Research and
Development Center

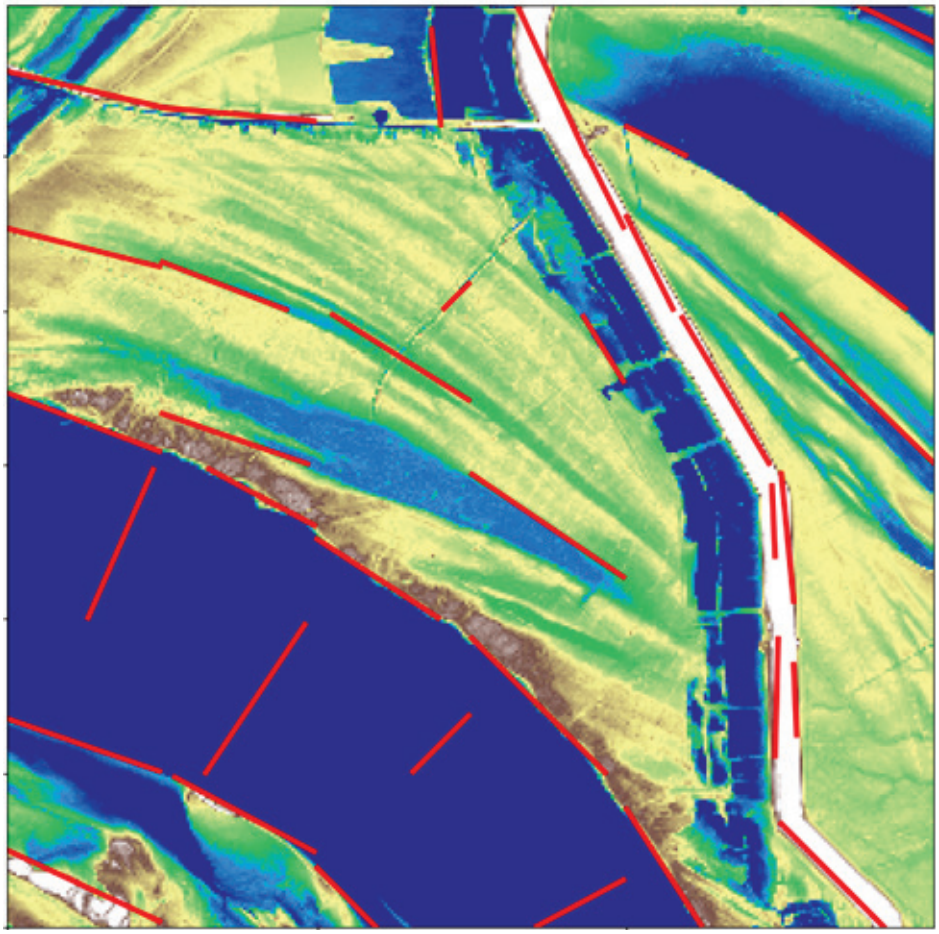
ERDC
ENGINEER RESEARCH & DEVELOPMENT CENTER

Flood and Coastal Systems R&D

Automated Characterization of Ridge-Swale Patterns Along the Mississippi River

Alicia D. Downard, Stephen N. Semmens, and
Bryant A. Robbins

April 2021



The U.S. Army Engineer Research and Development Center (ERDC) solves the nation's toughest engineering and environmental challenges. ERDC develops innovative solutions in civil and military engineering, geospatial sciences, water resources, and environmental sciences for the Army, the Department of Defense, civilian agencies, and our nation's public good. Find out more at www.erdc.usace.army.mil.

To search for other technical reports published by ERDC, visit the ERDC online library at https://erdclibrary.on.worldcat.org/discovery.

ERDC/GSL TR-21-13
April 2021

Automated Characterization of Ridge-Swale Patterns Along the Mississippi River

Stephen N. Semmens and Bryant A. Robbins

Geotechnical and Structures Laboratory
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180

Alicia D. Downard

Department of Geophysics, Undergraduate Student
Colorado School of Mines
1500 Illinois Street
Golden, CO 80401

Final report

Prepared for U.S. Army Corps of Engineers
Washington, DC 20314-1000

Under Flood & Coastal Civil Works Direct Program, Remote Monitoring & Sensing; Critical Infrastructure Protection and Resilience Program, Office of Homeland Security

Abstract

The orientation of constructed levee embankments relative to alluvial swales is a useful measure for identifying regions susceptible to backward erosion piping (BEP). This research was conducted to create an automated, efficient process to classify patterns and orientations of swales within the Lower Mississippi Valley (LMV) to support levee risk assessments. Two machine learning algorithms are used to train the classification models: a convolutional neural network and a U-net. The resulting workflow can identify linear topographic features but is unable to reliably differentiate swales from other features, such as the levee structure and riverbanks. Further tuning of training data or manual identification of regions of interest could yield significantly better results. The workflow also provides an orientation to each linear feature to support subsequent analyses of position relative to levee alignments. While the individual models fall short of immediate applicability, the procedure provides a feasible, automated scheme to assist in swale classification and characterization within mature alluvial valley systems similar to LMV.

Contents

Figures

Figures

Preface

This study was conducted for the Critical Infrastructure Protection and Resilience Program, Office of Homeland Security, U.S. Army Corps of Engineers, under the direction of the Program Manager, Ms. Yazmin Seda-Sanabria, and the Flood and Coastal Systems R&D Program under the direction of Dr. Julie Rosati.

The work was performed by the Geotechnical Engineering and Geosciences Branch (GS-G) of the Geosciences Division (GS), U.S. Army Engineer Research and Development Center, Geotechnical and Structures Laboratory (ERDC-GSL). At the time of publication, Mr. Christopher Price was Chief, GSG; Mr. James L. Davis was Chief, GS; and Dr. Michael K. Sharp, GST, was the Technical Director for Water Resources Infrastructure. The Deputy Director of ERDC-GSL was Mr. Charles W. Ertle II, and the Director was Mr. Bartley P. Durst.

The authors wish to acknowledge Dr. Joseph B. Dunbar of ERDC-GSL for both project assistance and inspiration. Dr. Dunbar originally conceived the idea of performing automated characterization of geomorphic features pertinent to sand boil formation from LiDAR (Light Detection and Ranging) as part of research projects on remote sensing. The authors, all working on various aspects of the problem presented, came together through a series of serendipitous events to produce the results presented.

COL Teresa A. Schlosser was the Commander of ERDC, and Dr. David W. Pittman was the Director.

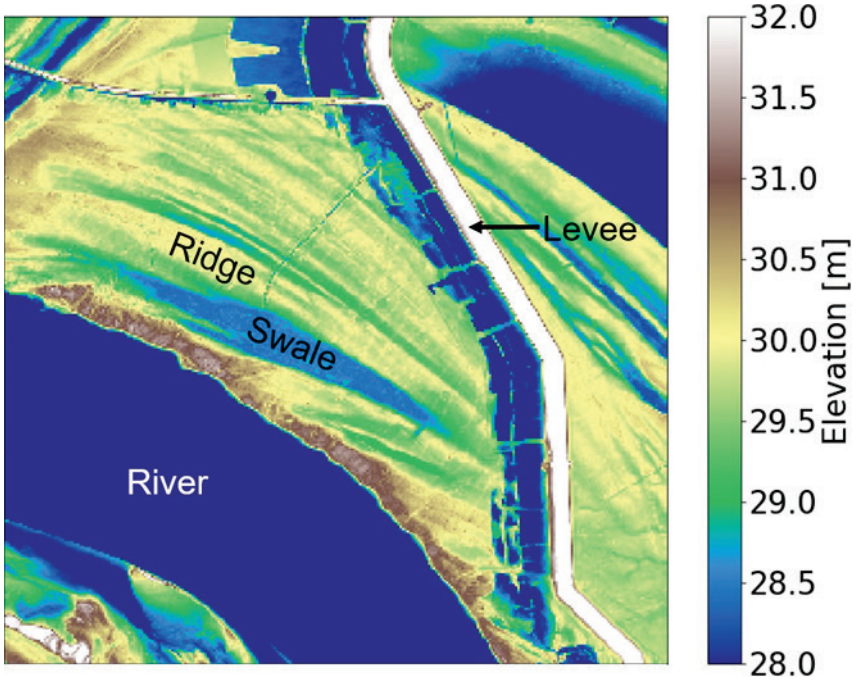
1 Introduction

1.1 Background

The Mississippi River and its tributaries form a drainage basin over 3,224,000 km² in size in the center of the continental United States (EPA 2016). Since the Flood Control Act of 1928, the U.S. Army Corps of Engineers (USACE) has maintained comprehensive flood control and navigation infrastructure along the river, including the construction and maintenance of levees for containing flood flows (USACE 2004).

Within the Lower Mississippi Valley (LMV), levee systems sit atop a heterogeneous assortment of materials deposited by the dynamic interaction of the river with the surrounding landscape. At the ground surface, material and depositional properties may manifest in distinct topographic expressions. One such pattern, referred to as ridge-swale topography, is common throughout the alluvial valley as curvilinear, approximately parallel undulations in the topography (Lobeck 1939). Ridges represent linear regions of higher elevation, while the depressions between ridges are called swales. An example of this pattern is shown in Figure 1.

Figure 1. A ridge-swale pattern along the Mississippi River identified on a digital elevation map.



Orientations of geologic features, including swales, relative to levee structures have been shown to contribute to the likelihood of sand boils during flood events (Mansur and Kaufman 1957; Kolb 1975). Sand boils are surficial deposits of foundation sand eroded and deposited by seepage from the levee foundation. Sand boils may lead to the occurrence of backward erosion piping (BEP), wherein shallow erosion channels propagate upstream through the levee foundation (Bonelli 2013), which is a primary failure mechanism for levee embankments. Left unattended, the erosion may continue to remove foundation material until the levee embankment can no longer be supported by the foundation and failure occurs. As early as 1956, USACE observed elevated occurrences of sand boils in certain geometries of swales (Mansur and Kaufman 1957). Multiple statistical studies indicate that deposits of channel fill in addition to swales likely contribute to the spatial distribution of sand boils and thus, BEP (Glynn et al. 2012; Semmens et al. 2017).

The influence of geology on BEP can likely be explained by the interaction between water and the physical properties of materials within the respective deposit. Swales and channel fills are characteristically composed of finer-grained material. Relative to the coarser material in surrounding deposits, swales and channel fill are known to focus, funnel, or otherwise significantly affect the flow of groundwater (Mansur and Kaufman 1957; Kolb 1975; Glynn et al. 2012). These deposits often contain more cohesive clay material, which better resists the erosional forces driving BEP.

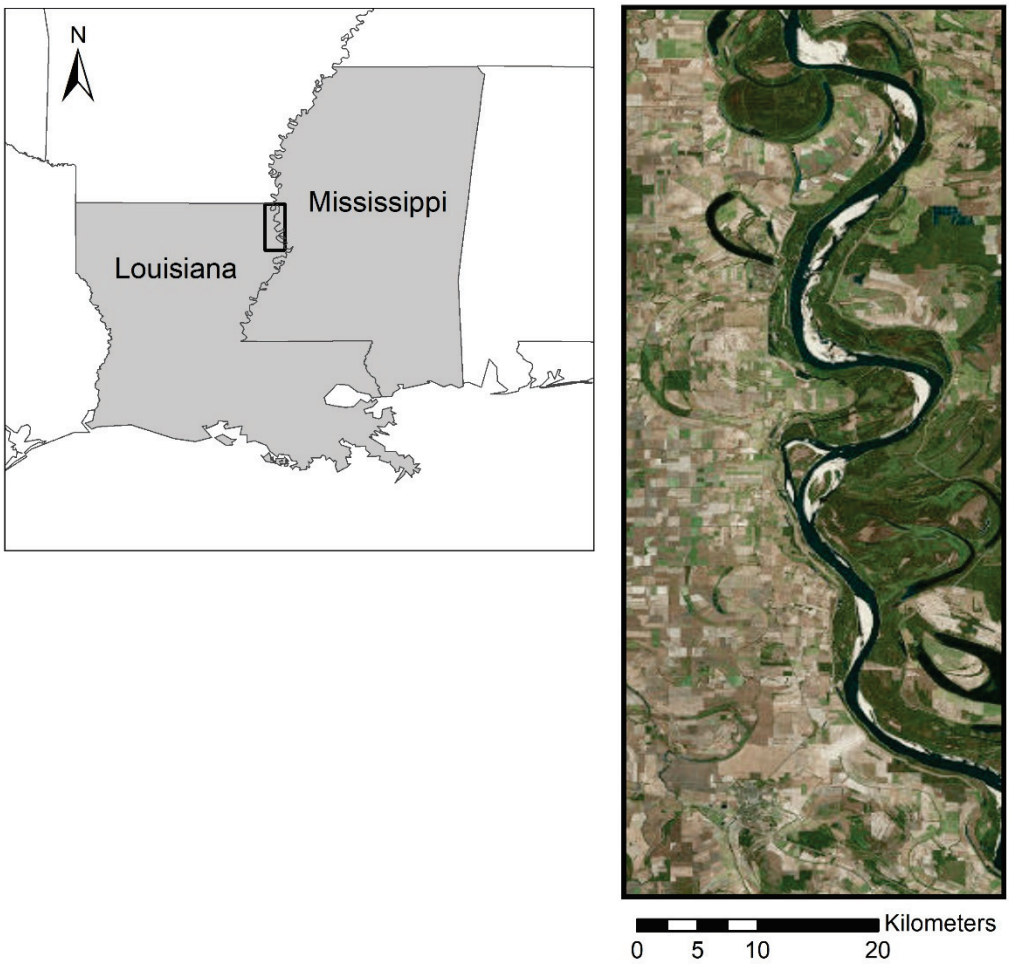
Given the apparent influence of swales on the spatial occurrence of BEP, creating a process to automate the identification of swale features and map their trends could provide an essential step toward efficient characterization of BEP hazard along levee systems. The following report provides a review of research performed toward the construction of such a process.

1.2 Study area

The study area consists of a 2,000 km^2 portion of LMV straddling the Mississippi River immediately south of the Arkansas border. A map of this region is shown in Figure 2. The region was selected for three primary reasons. First, it contains a significant number of swales for model training and assessment. Second, key data, such as digital elevation models (DEMs) and aerial imagery were readily available for the entire site. Third, the site contained a variety of other features (e.g., levees, oxbow lakes, the

Mississippi River, and farm plots), which provided additional topographic responses to challenge and assess the final model.

Figure 2. Aerial image of study area with reference to regional position.



1.3 Objective

The objective of this project is to investigate a method for the automatic classification and characterization of ridge-swale patterns along the Mississippi River. The following approach was used to investigate the objective:

1. Create a workflow for classification of ridge-swale patterns using DEMs.
2. Create a workflow for determination of angles of orientation between ridge-swale patterns and the corresponding levee.

3. Integrate these two workflows to create an automated, comprehensive procedure for the support of subsequent analysis of BEP as part of levee risk analyses.

1.4 Approach

The research objectives are addressed through development and application of both machine learning and image-processing techniques. First, the necessary DEM data are acquired. These data are then used to establish the training data set. Next, two machine learning algorithms (i.e., convolutional neural network and U-net) are implemented and assessed. The results are then passed through a series of image-processing steps to identify linear features and trends. Finally, the outputs are organized and assessed to determine the effectiveness of the workflow.

2 Procedure

2.1 Data acquisition

DEMs provide a useful data source for identifying ridge-swale systems given their unique pattern of topographic relief. For this study, DEMs are sourced from the USGS National Geospatial Program (USGS 2017).

The spatial resolution of the data set is 1/3 arc-sec (approximately 10 m), which is sufficient for the identification of smaller-scale features, such as ridges and swales. The elevation data are provided in meters and, over the conterminous United States, are referenced to the North American Vertical Datum of 1988 (NAVD 88). The vertical resolution of the data is 1 m, which is fine enough to distinguish the elevation differences in ridge-swale patterns.

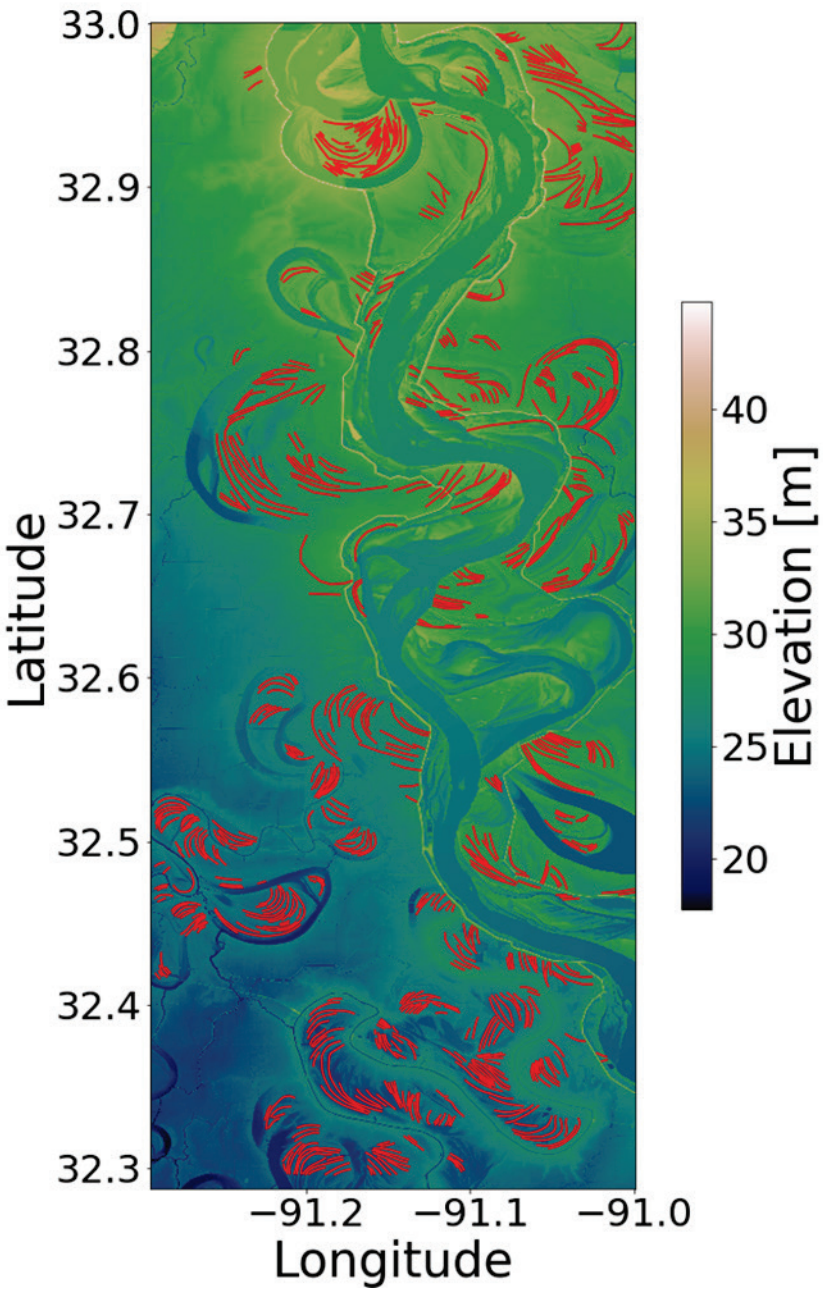
The information is then trimmed to the study area identified in Figure 2 to remove excess data and better focus on space bounding the Mississippi River and associated artificial levee structures.

2.2 Classification of ridge-swale patterns

2.2.1 Processing

To implement the classification scheme, a labeled data set is required. In this case, the labeled data set is derived by visual interpretation. The DEM is inspected for the characteristic undulating linear morphology of ridges and swales. Swales are then labeled by digitizing a line down the center of the swale trough. Figure 3 depicts 757 swales interpreted within the study area. Subsequent processing is performed on this data set to create data labels compatible with each of the classification schemes.

Figure 3. DEM of study area (Figure 2) containing interpreted swales (red lines).



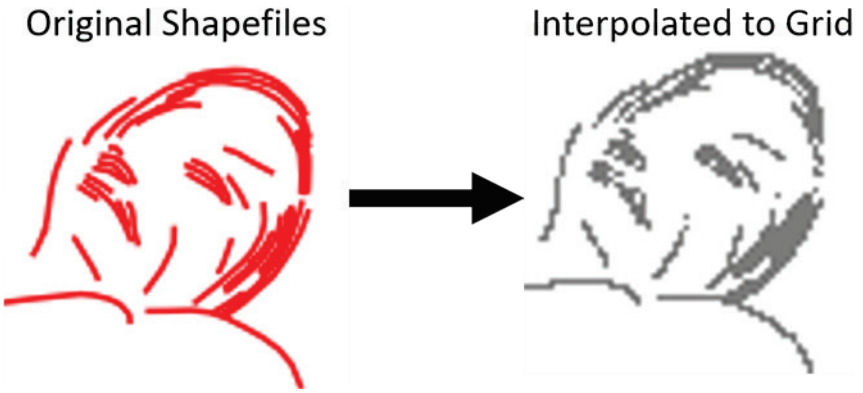
2.2.2 Classification using a convolutional neural network

The first machine learning classification algorithm investigated is a convolutional neural network (CNN). This is a type of deep-learning algorithm frequently implemented in image classification tasks because it is able to "successfully capture the spatial and temporal dependencies in an image through the application of relevant filters" (Saha 2018). The algorithm does so by assigning weights and biases to various features in

the image. It then updates these weights by comparing them to the class label of an image to produce a trained model.

CNNs perform image-wise classification; therefore, they require individual images with labels as their input data. Thus, the labeled DEM containing the 757 swales is made into individual images, each with an associated class label. To accomplish this, the polyline shapefiles are interpolated to map the location of the swale features to the associated grid values. This allows for nearly all the pixels in the DEM grid that contain the swale shapefile to have a corresponding point. Figure 4 shows an example of the interpolation.

Figure 4. A depiction of the interpolation of interpreted swales (left) represented as red lines versus corresponding discretized grid values (right) represented as individual pixels shown in gray.



Once the interpolation is achieved, the DEM and the interpolated swale grid are then subdivided into 2,232 sections. Only two class labels are used to automate the labeling of these images and avoid time-consuming manual labelling of every image: 0 for non-swale images, and 1 for swale-images. If the swale grid corresponding to one of the sub-divided DEM images contains at least one swale pixel, it is assigned a class label of 1. If the swale grid corresponding to the sub-divided DEM images contains no swale pixels, it is assigned a class label of 0. Various thresholds are also used to try and distinguish images with a greater number of swale pixels (e.g., setting the class label to 1 only if the image contained greater than 10 swale pixels) to see if this has an impact on prediction results.

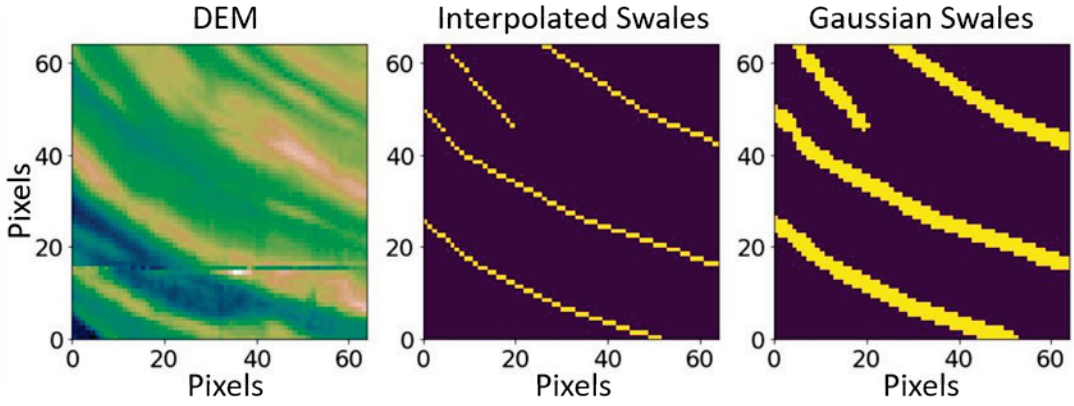
Once the DEM images are given class labels, a convolutional neural network is used to train a model. Results produced by this model are presented in Section 3.1.1.

2.2.3 Classification using a U-net

The second machine learning classification algorithm investigated is a U-net. The U-net architecture takes advantage of convolutional neural network model but implements it in a slightly different structure (Lamba 2019). In the case of the U-net, the result is pixel-wise classification, as opposed to image-wise classification of the traditional CNN architecture.

Because the U-net is designed to perform pixel-wise classification, each pixel of the DEM is assigned a class label. The resulting classes are contained in an image format, commonly referred to as a "mask." In this implementation, a swale mask is constructed in two steps. First, each of the interpolated grid values from the shapefile is assigned a class label of 1. Second, a Gaussian smoothing filter with thresholding is applied to correct for potential gaps in the mask or misalignments with the swale feature. The result is more continuous, fuller-coverage mask labels. An example of this process is shown in Figure 5.

Figure 5. DEM (left) and associated masks after swale interpretation (middle) and Gaussian filter with thresholding (right).



Once these masks are created, the U-net algorithm is applied to train a model. The results are discussed in Section 3.1.2.

2.3 Characterization of ridge-swale patterns

While the previous sections outline approaches for automated classification of swales, it is also of interest to automatically detect the swale pattern orientation in relation to corresponding levees. This section describes a proposed procedure for such an analysis.

2.3.1 Highlighting linear features

As ridge-swale patterns can appear as linear features in DEMs, the DEM is first divided into smaller sub-images. Once the larger region is subdivided, a series of three processing steps is applied to the data. Because ridges are relatively higher elevations than swales, taking the gradient of an image highlights the boundary between adjacent ridges and swales, as this constitutes a relatively rapid change in elevation. Thus, the first step is to take both horizontal and vertical derivatives of each image. From these values, the magnitude of the gradient is computed using the following relationship:

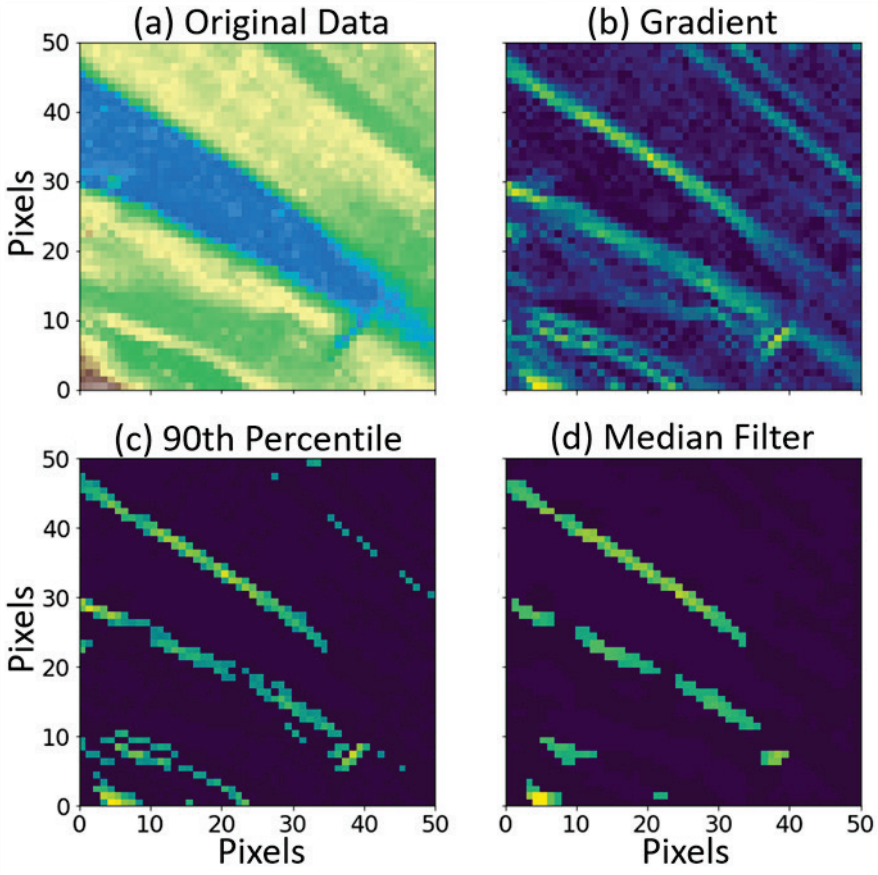
$$\|\nabla \boldsymbol{G}\| = \sqrt{\boldsymbol{G}_x^2 + \boldsymbol{G}_y^2} \tag{1}$$

where:

$\|\nabla \boldsymbol{G}\|$ = magnitude of the gradient
$\boldsymbol{G}_x$ = horizontal derivative of image
$\boldsymbol{G}_y$ = vertical derivative of image

An example of a gradient calculation performed on a ridge-swale pattern is shown in Figure 6. Figure 6(a) shows the original section of elevation data containing a ridge-swale pattern. Figure 6(b) shows the magnitude of the gradient of the original data.

Figure 6. A depiction of the spatial domain processing steps implemented on the DEM data set.



The second processing step involves removing low magnitude values of the gradient. This eliminates data points representing areas of relatively small changes in elevation to highlight the undulating ridge-swale topography. This is accomplished by calculating the 90th percentile value of the magnitude of the gradient and assigning all data points that fall below this threshold a value of zero. An example of this can be seen in Figure 6(c). This results in areas of large gradients being preserved.

Though the linear features are significantly more pronounced after removing the bottom 90th percentile of gradient values, there are still areas of discontinuity between neighboring data points. Ridge-swale patterns are typically characterized by distinct, continuous boundaries between areas of high and low elevation. Thus, in the third processing step, a median filter is applied to better distinguish the continuous gradient values in the data and further remove areas of discontinuous gradient values. An example of these filtering results is shown in Figure 6(d).

2.3.2 Application of the Hough transform

Once the linear features are adequately distinguished, the Hough transform is applied. This transform is commonly used for feature detection in image-processing applications and is applied in this workflow as a method of identifying linear features in images.

The Hough transform makes use of the normal parameterization of a line, which makes it particularly useful for this application. In this parameterization, a line can be described in the xy-plane by its distance (ρ) and angle (θ) parameters through the following relationship:

$$\boldsymbol{\rho = x \cos \theta + y \sin \theta} \quad (2)$$

where:

$\boldsymbol{\rho}$ = distance from origin to nearest point on the line
$\boldsymbol{x}$ = horizontal position
$\boldsymbol{y}$ = vertical position
$\boldsymbol{\theta}$ = angle between the horizontal x-axis and the line connecting the origin and the nearest point

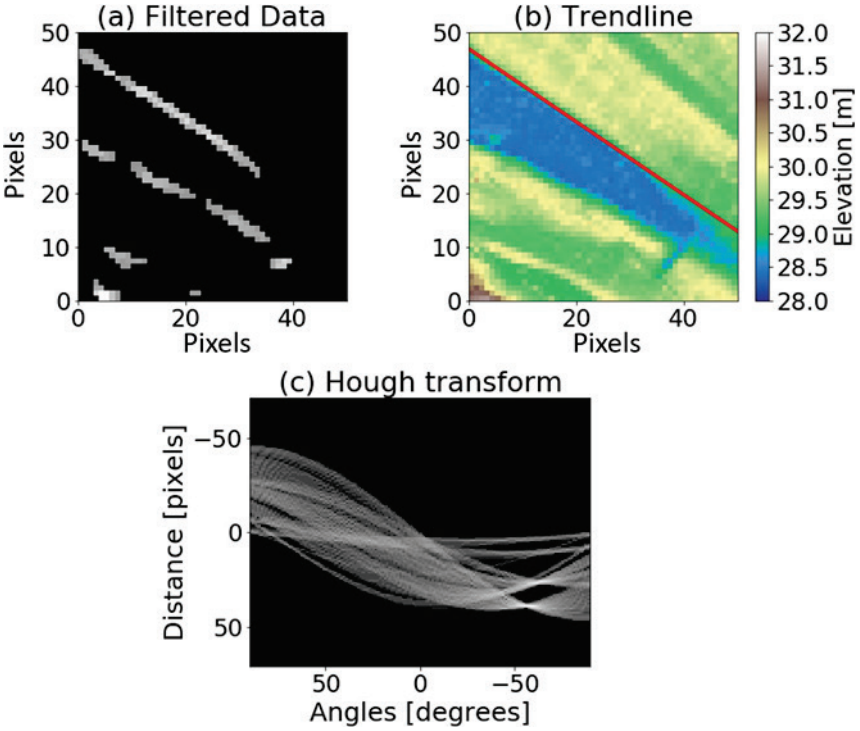
In practice, the transform essentially loops through various angles in an image and returns the (ρ,θ) parameters of the line that yield the dominant linear trend. This allows for the determination of angles of linear features in the processed DEM sub-images.

The implementation of the transform is straightforward in Python. For this implementation, two functions from the `skimage` processing library are used: `hough_line` and `hough_line_peaks`. With these functions, the Hough transform is applied and the maximum intensity trend line determined. The trend line is then plotted on the original data.

The Hough transform provides two key benefits towards the project goals. First, plotting trend lines provides a visual output that may be compared against the original swale interpretation and DEM. Second, the transform returns an angle for use in subsequent spatial and orientation analyses between swales and other elements, such as a levee structure. An example of the application of the Hough transform is shown in Figure 7. Figure 7(a) shows data filtered using the processing steps described in Figure 6. Figure 7(c) depicts the filtered data after applying the Hough transform.

The data are plotted as angle versus distance. Note that areas of intersection indicate the most prevalent orientation of linear features observed in the data. Figure 7(b) shows the maximum trend line determined by the highest intensity angle in Figure 7(c). The trend line follows the ridge-swale boundary almost exactly.

Figure 7. An example of the Hough transform applied to a filtered section of data containing a ridge-swale pattern.



3 Results and Discussion

3.1 Results

3.1.1 CNN classification

Classification using a CNN did not yield meaningful results. The algorithm delivered appreciable accuracy above 70 percent but on further investigation, the model was predicting only a single class and was incapable of distinguishing images that contained swales. One explanation is that the number of images with a 0 class label (containing no swales) was significantly larger than the number of images with a 1 class label (containing swales). Thus, the highest accuracy the model could produce was achieved with predicting all images as class 0.

This issue could be investigated and addressed through a reduction in the number of class 0, non-swale images used to train the model. For example, the study area could be further clipped to remove excessive non-swale regions farther from the banks of the Mississippi River. This may produce more meaningful results. However, this was not tested because the U-net approach proved a more favorable alternative to the CNN method.

3.1.2 U-net classification

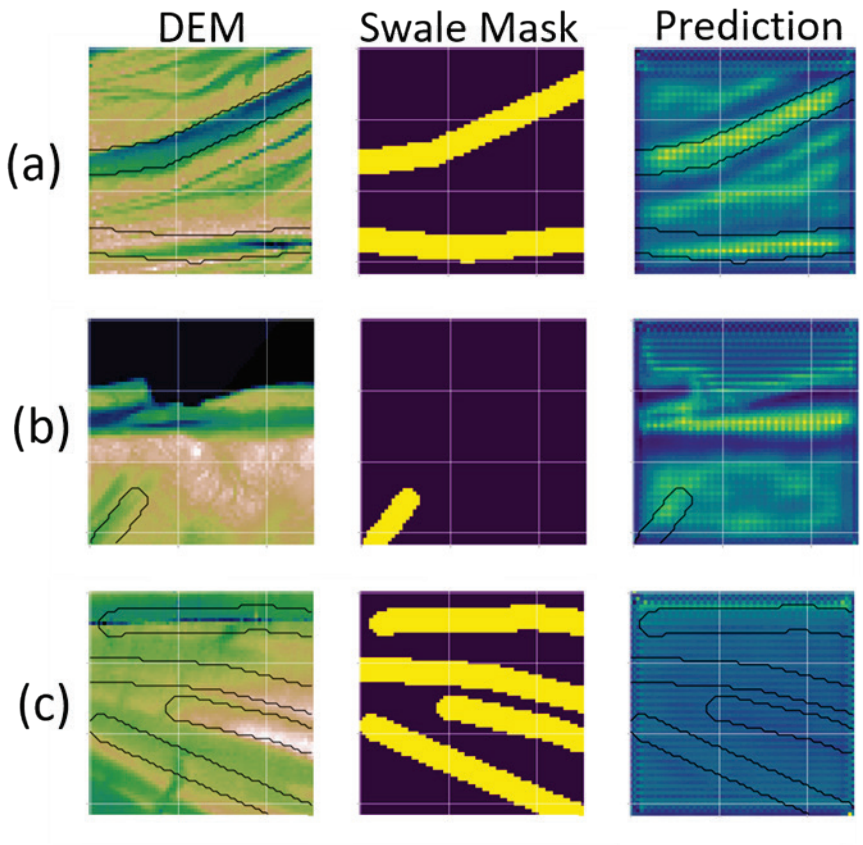
Using a U-net approach was determined to be more favorable than a CNN approach for swale classification. One reason for this was the aspect of pixel-wise classification because it provides the ability to not only identify an area in which a ridge-swale pattern exists, but to also identify where exactly that pattern is located within an image. Additionally, the model trained using a U-net approach yielded results that indicated an ability to distinguish swale features from non-swale features, whereas the model trained using a CNN did not distinguish features. However, despite improved results over the CNN model, the U-net model results were still not determined to be sufficient for operational use for reasons discussed in the paragraphs that follow.

The final U-net model prediction results were varied. Three general cases were observed. In the first case, it appeared that the model was able to accurately determine where swale features were in an image. Masks for these images were reasonably well-aligned with the corresponding swale features. In the second case, the model was unable to identify regions corresponding to swale masks, instead identifying

non-swale features that exhibited elements of morphology similar to the ridge-swale pattern but were not interpreted as such. In the third case, swale and non-swale features were not identified at all, and no distinguishable prediction was made by the model.

An example of each case is shown in Figure 8. The panels in the left column represent DEM images. The panels in the middle column contain swale masks with yellow pixels representing a class label of 1 (swale pixel) and purple pixels representing a class label of 0 (non-swale pixel). The panels in the right column show the prediction results from the trained model with yellow pixels representing relatively higher probabilities of being swales and blue and purple pixels representing lower probabilities. Row (a) depicts promising results, where the yellow regions of higher intensity correspond well with the swale masks and areas of low elevation in the DEM. Row (b) shows slightly less favorable results. The horizontal feature in the image appears to have been recognized by the model, but it failed to predict the presence of the interpreted swale in the bottom right corner of the image. Row (c) shows no distinguishable results. Note that the masks created do not exactly align with distinguishable swale features in the actual DEM image. This could potentially contribute to the lower prediction accuracy observed. It is possible that the DEM accuracy was also limited by the elevation contrast. Convolution operations would further blur these features, resulting in limited contrast for detection neurons in the algorithm. With higher resolution DEM data, it is possible that this approach would work quite well.

Figure 8. Three different prediction results from the model trained with a U-net with the original DEM (left column), swale masks (middle column), and prediction results from the trained model (right column).



The procedure used to create the image masks likely had a negative effect on result accuracy. A clear instance of this can be observed in Figure 8(c). In the DEM, it is unclear whether a swale feature exists (compare to the masked features in Figure 8[a]) despite four interpreted swale features indicated in the image mask. It may also cause some features that are swales to be misidentified as non-swales. This unfavorable effect is a result of the interpolation and filtering used to create the image masks. In the original labeling of the DEM, swales were identified using shapefiles that are not directly transferable to image masks. Thus, the "rough-but-fast" method used to create masks from the shapefiles themselves likely limited the accuracy of the results.

3.1.3 Ridge-swale orientations

The final step of the characterization workflow involved assigning a measure of orientation to identified swale features. Figure 9 depicts a region that was divided into 36 sub-images to complete the full characterization

workflow. The results shown in Figures 9 and 10 are promising. First, the image displays that many of the ridge-swale boundaries are identified, along with the associated angles. The levee was also distinguished exceptionally well. This makes sense as the algorithm is designed to identify dominant linear trends. As a result, if multiple linear features, such as a swale and levee structure, occur within a sub-image, only the dominant linear trend will be identified. Minimizing sub-image size or masking unwanted features exhibiting strong linear trends can reduce suppression of swale responses.

Figure 9. DEM where the characterization workflow has been applied to subsections to plot the dominant trend lines (red).

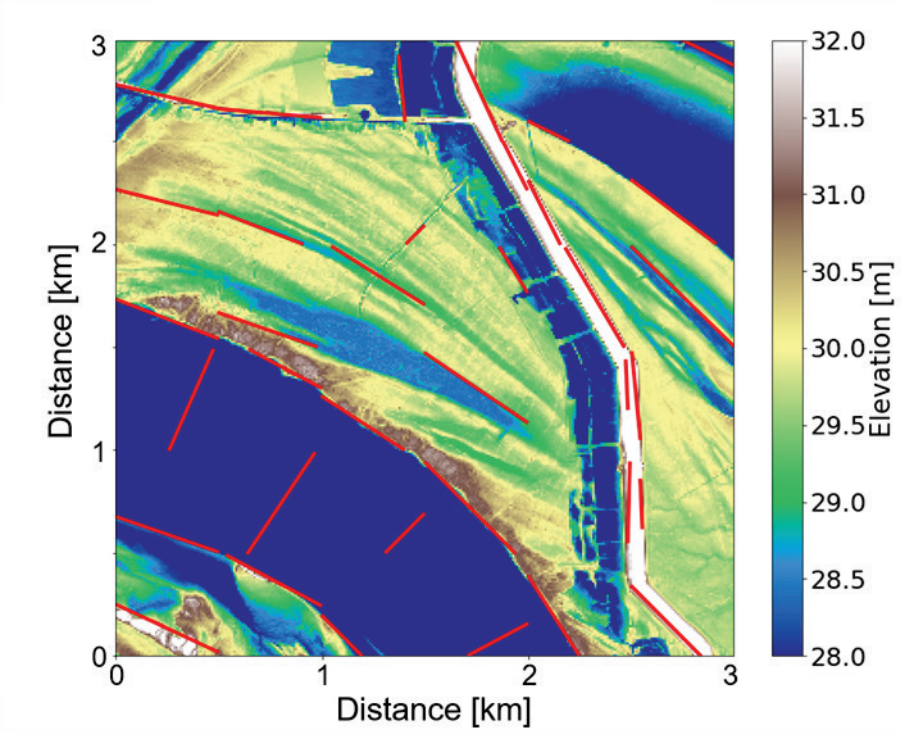
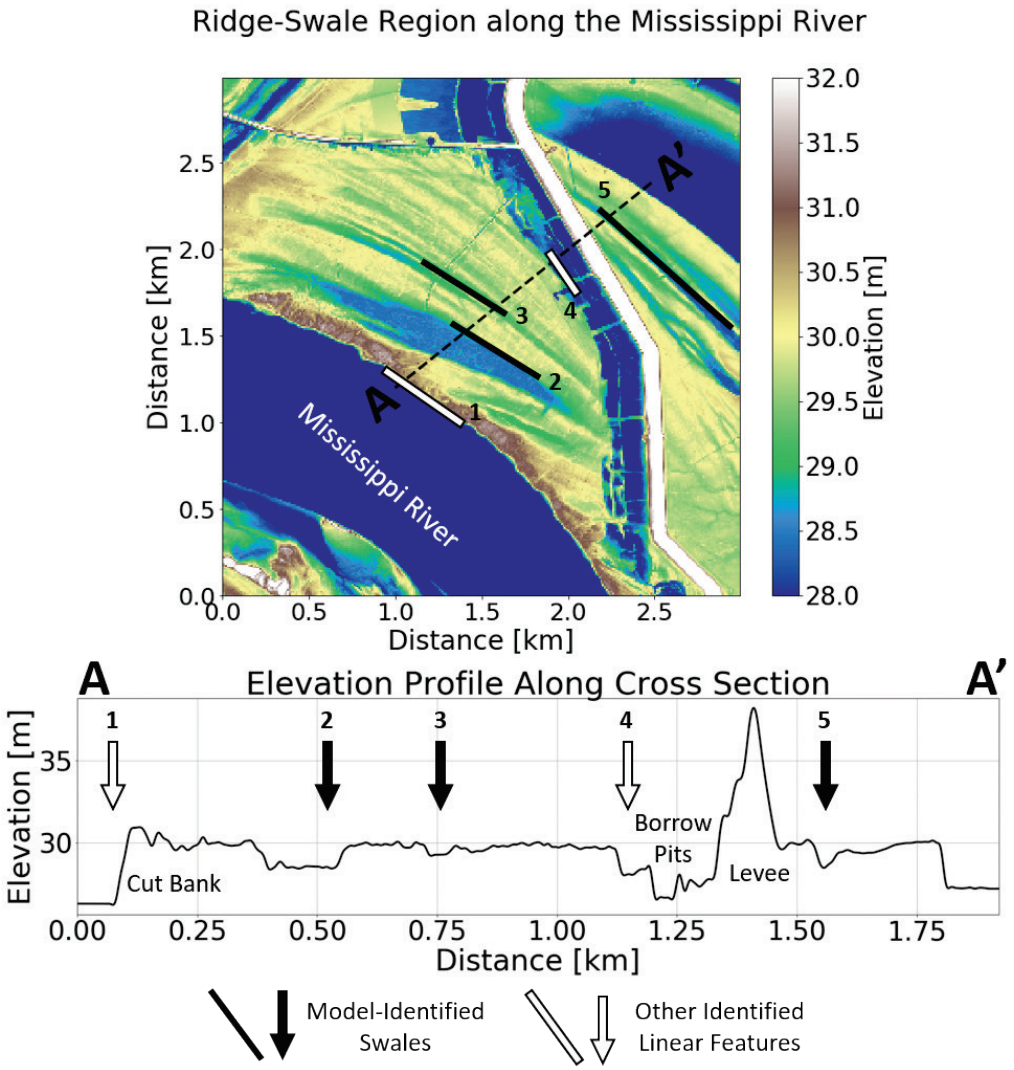


Figure 10. DEM and cross section with intersecting dominant linear trends.



This result has a practical application in helping determine the angles of orientation between swales and adjoining levees. Because this workflow was able to plot the trend lines and return angles of swales and levees, determining the orientations between these features should be straightforward.

Another interesting response from Figure 9 is the trend lines plotted within the river and along the riverbanks. These trend lines do not provide any meaningful information but do help in highlighting a key characteristic of this workflow. The methodology does not have the ability to interpret features; it merely plots the dominant linear trend observed in the data. Though it can "find" the boundaries between ridges and swales, this is a result of the processing steps integrated into the workflow. To

become more useful in determining orientations of ridge-swale patterns specifically, the classification scheme must be integrated to initially identify where these patterns exist in a DEM.

4 Conclusions and Recommendations

4.1 Conclusions

The Mississippi River is the main channel in part of a large, dynamic drainage basin in the center of the continental United States. The dynamic nature of this system, along with its many tributaries, lends itself to frequent flooding events. Without the current man-made structures in place, one of which is a levee, these flooding events would devastate local infrastructure along the river. Thus, it is essential that the integrity of these structures be maintained. Current methods in place to evaluate maintenance primarily include manual inspections, which are time-consuming and inefficient. Though these manual inspections provide valuable, in-depth information, they generally assess only where maintenance is currently required and do not anticipate where maintenance may be needed in the future. Thus, there is a need for a more automated, predictive method to determine possible areas of failure in these structures before a failure occurs.

It has been shown that the orientation of ridge-swale patterns with levees can lead to sand boils and possibly BEP, which can compromise the integrity of levees and possibly lead to failure. Thus, the successful implementation of the outlined approach to automatically classify and characterize ridge-swale patterns in DEMs would provide an efficient method for determining a key predictor of levee failure, helping to improve the current measures already in place and to reduce the harmful effects of flooding events along the Mississippi River.

While the results of this project are promising, in their current state the workflows are not yet operational. The first step in the approach of the project was to create a workflow for the classification of ridge-swale patterns using DEMs. A U-net pixel-wise classification scheme was used to train a model, the accuracy of which would need to be improved before it can be implemented for practical use. However, some of the prediction results showed the ability of the model to distinguish between swale and non-swale features. Further tuning of the model and input data would likely increase its prediction accuracy, possibly to an appropriate level for operational implementation. The second step of the project was to create a workflow for determination of angles of orientation between ridge-swale patterns and the corresponding levee. The workflow created to

accomplish this proved effective in identifying linear trends in DEMs and can return angles of the associated trend lines. In particular, the workflow was able to identify linear trends of both levees and ridge-swale features in DEMs, a useful step in fulfilling this objective; in its current form, the workflow does not return angles of orientation between features. The third step was to integrate the two workflows to create an automated, comprehensive procedure for the support of subsequent BEP hazard analysis. With the successful implementation of the classification and characterization workflows along with the proper integration with engineering applications, this objective is attainable. Recommendations for how to accomplish this are discussed in the following section.

4.2 Recommendations

One of the key faults in the implementation of the U-net was the process of creating image masks. Certain features were masked that did not represent swales, while other swale features were not masked. Implementing a more thorough labeling of swale features would aid in the improvement of this process. This could be achieved using polygon instead of polyline shapefiles in ArcGIS to more accurately and completely mask swale features. Another area of improvement could come from the integration of additional data types with the DEM data. Using normalized difference vegetation index (NDVI) or other supplemental data to further assist in either identifying swale or non-swale features may be another avenue worth pursuing.

The angle characterization workflow needs development. While the workflow can determine feature orientations, the outputs do not yet include angles of orientations between swale and levee features and are not optimized for use by a specific end-user. Implementing the workflow as an ArcGIS tool with associated output objects would be a step toward increasing the ease of use and accessibility to geologists and engineers who are likely to be the ones performing posterior analysis.

Once the classification and characterization are working as intended, the final step would be integrating them. The machine-learning classifier would identify areas in a DEM that resemble ridge-swale features, then these areas would be directly fed into the angle characterization workflow. Finally, the outputs could be passed on in an appropriate format for further spatial assessment.

References

Bonelli, S (ed). 2013. *Erosion in geomechanics applied to dams and levees*. New York: Wiley.

Glynn, E., M. Quinn, and J. Kuszmaul. 2012. *Predicting piping potential along middle Mississippi River levees*. 6th International Conference on Scour and Erosion. Paris: Société Hydrotechnique de France.

Kolb, C. R. 1975. *Geologic control of sand boils along Mississippi River levees*. AEWES-MISC-PAPER-S-75-22. Vicksburg, MS: U.S. Army Engineer Waterways Experiment Station.

Lamba, H. 2019. Understanding semantic segmentation with UNET: A salt identification case study. https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47.

Lobeck, A. K. 1939. *Geomorphology, an introduction ot the study of landscapes*. New York: McGraw-Hill Book Company.

Mansur, C. I., and R. I. Kaufman. 1957. Underseepage, Mississippi River levees, St. Louis District. *Transactions of the American Society of Civil Engineers* 985-1008.

Saha, S. 2018. A comprehensive guide to convolutional neural networks - the ELI5 way. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

Semmens, S. N., W. Zhou, B. K. Wesenbeeck, and P. M. Santi. 2017. Application of multiple criteria decision making model for evaluation of levee sustainability. *Environmental and Engineering Geoscience* 65-78.U.S. Army Corps of Engineers (USACE). 2004. The Mississippi River and tributaries project. https://biotech.law.lsu.edu/maps/mrtp/mrtp.htm.

U.S. Environmental Protection Agency (EPA). 2016. The Mississippi/Atchafalaya River Basin (MARB). https://www. epa.gov/ms-htf/mississippiatchafalaya-river-basin-marb.

U.S. Geological Survey (USGS). 2017. National Geospatial Program. https://www.usgs.gov/core-science-systems/national-geospatial-program.

Appendix A

A.1 CNN classification

```
import numpy as np import pandas as pd
import matplotlib.pyplot  as  plt from matplotlib
import cm
from osgeo import gdal import geopandas as gpd
import  shapefile as shp # Requires the pyshp package
from scipy.interpolate import interp1d

from keras.models import Sequential
from  keras.layers  import  Dense,  Conv2D,  Flatten

geo =
gdal.Open('./USGS_NED_13_n33w092_IMG/USGS_NED_13_n33w0
92_IMG.img') arr = geo.ReadAsArray()

sf =
shp.Reader('./FullInterp/Swales_Interpreted_Full_SS.sh
p') npoints = 0
for shape in sf.shapeRecords():
x = [i[0] for i in shape.shape.points[:]]  y = [i[1]
for i in shape.shape.points[:]]



npoints += len(x)
swale_grid = np.nan  *  np.empty((np.shape(AOI)))  xc
= xAOI[0,:]
yc  = yAOI[:,0]
x_idx = np.arange(0,len(xc),1) y_idx =
np.arange(0,len(yc),1)

for x,y in zip(all_swales[:,0],all_swales[:,1]):

xPos = x_idx[np.abs(x - xc) == np.min(np.abs(x - xc))]
yPos = y_idx[np.abs(y - yc) == np.min(np.abs(y - yc))]

swale_grid[yPos,xPos] = 1
```

```
npix = 10
[xdim, ydim] = np.shape(AOI) nx = int(xdim / npix)
ny  =  int(ydim  / npix)
xidx = np.arange(0,xdim,npix) yidx =
np.arange(0,ydim,npix)

dataList = []

for iy in range(ny-1):

smallerArea = AOI[ :, yidx[iy]:yidx[iy+1] ] subArrays
=  np.split(smallerArea,  nx,  0) swaleArea =
swale_grid[ :, yidx[iy]:yidx[iy+1] ] subSwales =
np.split(swaleArea, nx, 0)
for swales,data,ix  in
zip(subSwales,subArrays,range(nx)): label = 0
if np.sum(~np.isnan(swales)) > 0: label = 1




dataList.append((data,label))

#  create random  indeces to split data into train and
test   nd = len(dataList)
nt = int(np.floor(0.10 * nd)) # number of test data
nv = int(np.floor(0.10 * nd)) # number of validation
data  rand_idx = random.sample(range(nd), nt+nv)
idx_train = np.ones(nd) idx_test =  np.zeros(nd)
idx_val = np.zeros(nd)

idx_train[rand_idx] = 0
idx_test[rand_idx[0:nt]] = 1
idx_val[rand_idx[nt:]] = 1
n_train, n_test, n_val = int(np.sum(idx_train)),
int(np.sum(idx_test)), int(np.sum(idx_val))
x_train,y_train,x_test,y_test,x_val,y_val =
[],[],[],[],[],[] for i in range(nd):
x  =  dataList[i]
```

```
if idx_train[i]==1: x_train.append(x[0].ravel())
y_train.append(x[1])

elif idx_test[i]==1: x_test.append(x[0].ravel())
y_test.append(x[1])

elif idx_val[i]==1: x_val.append(x[0].ravel())
y_val.append(x[1])

x_train = np.array(x_train).reshape(n_train, npix,
npix, 1)



x_test    = np.array(x_test).reshape(n_test, npix,
npix, 1) x_val = np.array(x_val).reshape(n_val, npix,
npix, 1)
y_train, y_test, y_val = np.array(y_train),
np.array(y_test), np.array(y_val)

#create model
model = Sequential() #add model layers
model.add(Conv2D(64, kernel_size=3, activation='relu',
input_shape=(npix,npix,1))) model.add(Conv2D(32,
kernel_size=3, activation='relu'))
model.add(Flatten())
model.add(Dense(2, activation='softmax'))

#compile model using accuracy to measure model
performance model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy',
metrics=['accuracy'])

#train the model
model.fit(x_train, y_train, validation_data=(x_test,
y_test), epochs=3)

# make sure we didn't overfit, calculate validation
loss and validation accuracu val_loss, val_acc =
model.evaluate(x_test, y_test)
```

```
print(val_loss, val_acc)

predictions = model.predict([x_test]) # gives
probability distributions
```

A.2 U-net classification

```
import numpy as np import pandas as pd
import matplotlib.pyplot as plt from osgeo import gdal
from matplotlib import cm import geopandas as gpd
import  shapefile as shp # Requires the pyshp package
from scipy.interpolate import interp1d
from scipy.ndimage import gaussian_filter




# machine learning libraries import os
import random

from tqdm import tqdm_notebook, tnrange from itertools
import chain
from skimage.io import imread, imshow,
concatenate_images from skimage.transform import
resize
from skimage.morphology import label
from sklearn.model_selection import train_test_split
import tensorflow as tf
from keras.models import Model, load_model
from keras.layers import Input, BatchNormalization,
Activation, Dense,  Dropout  from keras.layers.core
import Lambda, RepeatVector, Reshape
from keras.layers.convolutional import Conv2D,
Conv2DTranspose from keras.layers.pooling import
MaxPooling2D, GlobalMaxPool2D from keras.layers.merge
import concatenate, add
from keras.callbacks import EarlyStopping,
ModelCheckpoint, ReduceLROnPlateau from
keras.optimizers import Adam
```

```
from keras.preprocessing.image import
ImageDataGenerator, array_to_img, img_to_array,
load_img

geo =
gdal.Open('./USGS_NED_13_n33w092_IMG/USGS_NED_13_n33w0
92_IMG.img') arr = geo.ReadAsArray()

bounds = pd.DataFrame(np.array([[-92.0005555556937, -
90.9994444453043,
33.0005555550957,31.9994444447062]]),
columns=['west', 'east', 'south', 'north'])
xcoord     =
np.linspace(bounds['west'].values[0],bounds['east'].va
lues[0],10812) ycoord  =
np.linspace(bounds['south'].values[0],bounds['north'].
values[0],10812) [XC,YC] = np.meshgrid(xcoord, ycoord)

xAOI = XC[0:7680, 7612:] yAOI = YC[0:7680, 7612:]



AOI = arr[0:7680, 7612:]
sf =
shp.Reader('./FullInterp/Swales_Interpreted_Full_SS.sh
p') # determine the difference between adjacent points
in grid
dx = np.abs(xAOI[1,1] - xAOI[1,0])

xc = np.zeros(1) yc = np.zeros(1)

for shape in sf.shapeRecords():
x = [i[0] for i in shape.shape.points[:]]  y = [i[1]
for i in shape.shape.points[:]]

# interpolation
f = interp1d(x, y)
xnew = np.arange(np.min(x), np.max(x), dx) ynew =
f(xnew)
```

```
xc = np.append(xc, xnew) yc = np.append(yc, ynew)

xc = np.delete(xc, 0) yc =  np.delete(yc, 0)

all_swales = np.zeros([len(xc),2]) all_swales[:,0] =
np.array(xc) all_swales[:,1] = np.array(yc)

# preallocate grid
swale_grid = np.zeros(np.shape(AOI))

xc  = xAOI[0,:]
yc  = yAOI[:,0]
x_idx = np.arange(0,len(xc),1) y_idx =
np.arange(0,len(yc),1)



for x,y in zip(all_swales[:,0],all_swales[:,1]):

xPos = x_idx[np.abs(x - xc) == np.min(np.abs(x - xc))]
yPos = y_idx[np.abs(y - yc) == np.min(np.abs(y - yc))]

swale_grid[yPos,xPos] = 1

# set the number of desired pixels for each dimension
of image  npix = 64
# get sizes of x and y dimensions [xdim, ydim] =
np.shape(AOI)
# set the number  of subdivisions of the larger DEM
along each dimension       nx = int(xdim / npix)
ny  =  int(ydim  / npix)
# create dummy index to use later xidx =
np.arange(0,xdim,npix) yidx = np.arange(0,ydim,npix)

dataList = [] maskList = []

for iy in range(ny-1):

smallerArea = AOI[ :, yidx[iy]:yidx[iy+1] ] subArrays
=  np.split(smallerArea,   nx,   0) swaleArea =
```

```
swale_grid2[ :, yidx[iy]:yidx[iy+1] ] subSwales =
np.split(swaleArea, nx, 0)

for swales,data,ix in
zip(subSwales,subArrays,range(nx)):
# do not include images that don't have any swale
pixels  if np.sum(swales) == 0:
continue

# Use gaussian filter to create better masks
gaussian_swales = gaussian_filter(swales, sigma=1)
gaussian_swales[gaussian_swales > 0.1] = 1
gaussian_swales[gaussian_swales    <    0.1]    =  0



dataList.append(data.reshape(npix,npix,1))
maskList.append(gaussian_swales.reshape(npix,npix,1))

X  =   np.array(dataList) y = np.array(maskList)

### NOTE: The following code was adapted from
# https://www.depends-on-the-definition.com/unet-
keras-segmenting-images/ # Split train and valid
X_train, X_test, y_train, y_test = train_test_split(X,
y, test_size=0.15, random_state=2018)

# Set some parameters im_width = 64
im_height = 64
border = 5

def conv2d_block(input_tensor, n_filters,
kernel_size=3, batchnorm=True): # first layer
x = Conv2D(filters=n_filters,
kernel_size=(kernel_size, kernel_size),
kernel_initializer="he_normal",
padding="same")(input_tensor) if batchnorm:
x  =  BatchNormalization()(x) x =
Activation("relu")(x)
```

```
# second layer
x = Conv2D(filters=n_filters,kernel_size=(kernel_size,
kernel_size),
kernel_initializer="he_normal",padding="same")(x)
if batchnorm:
x  =  BatchNormalization()(x) x     =
Activation("relu")(x) return x

def get_unet(input_img, n_filters=16, dropout=0.5,
batchnorm=True): # contracting path
c1 = conv2d_block(input_img, n_filters=n_filters*1,
kernel_size=3,



batchnorm=batchnorm)
p1 = MaxPooling2D((2, 2)) (c1) p1 =
Dropout(dropout*0.5)(p1)

c2 =  conv2d_block(p1,  n_filters=n_filters*2,
kernel_size=3, batchnorm=batchnorm)
p2 = MaxPooling2D((2, 2)) (c2) p2 =
Dropout(dropout)(p2)

c3 =  conv2d_block(p2,  n_filters=n_filters*4,
kernel_size=3, batchnorm=batchnorm)
p3 = MaxPooling2D((2, 2)) (c3) p3 =
Dropout(dropout)(p3)

c4 =  conv2d_block(p3,  n_filters=n_filters*8,
kernel_size=3, batchnorm=batchnorm)
p4 = MaxPooling2D(pool_size=(2, 2))  (c4) p4 =
Dropout(dropout)(p4)

c5 =  conv2d_block(p4,  n_filters=n_filters*16,
kernel_size=3, batchnorm=batchnorm)

# expansive path
u6 = Conv2DTranspose(n_filters*8, (3, 3), strides=(2,
2), padding='same') (c5)
```

```
u6 =  concatenate([u6,  c4]) u6 = Dropout(dropout)(u6)
c6 = conv2d_block(u6, n_filters=n_filters*8,
kernel_size=3, batchnorm=batchnorm)

u7 = Conv2DTranspose(n_filters*4, (3, 3), strides=(2,
2), padding='same') (c6)
u7 =  concatenate([u7,  c3]) u7 = Dropout(dropout)(u7)
c7 = conv2d_block(u7, n_filters=n_filters*4,
kernel_size=3, batchnorm=batchnorm)

u8 = Conv2DTranspose(n_filters*2, (3, 3), strides=(2,
2), padding='same') (c7)



u8 =  concatenate([u8,  c2]) u8 = Dropout(dropout)(u8)
c8 = conv2d_block(u8, n_filters=n_filters*2,
kernel_size=3, batchnorm=batchnorm)

u9 = Conv2DTranspose(n_filters*1, (3, 3), strides=(2,
2), padding='same') (c8)
u9  =  concatenate([u9,  c1],  axis=3) u9 =
Dropout(dropout)(u9)
c9 = conv2d_block(u9, n_filters=n_filters*1,
kernel_size=3, batchnorm=batchnorm)

outputs = Conv2D(1, (1, 1), activation='sigmoid') (c9)
model = Model(inputs=[input_img], outputs=[outputs])
return model

input_img = Input((im_height, im_width, 1),
name='img')
model = get_unet(input_img, n_filters=16,
dropout=0.05, batchnorm=True)

model.compile(optimizer=Adam(),
loss="binary_crossentropy",  metrics=["accuracy"])
model.summary()

callbacks = [
```

```
EarlyStopping(patience=10, verbose=1),
ReduceLROnPlateau(factor=0.1, patience=3,
min_lr=0.00001, verbose=1), ModelCheckpoint('model-
tgs-salt.h5', verbose=1, save_best_only=True,
save_weights_only=True)]
results = model.fit(X_train, y_train, batch_size=32,
epochs=5, callbacks=callbacks,
validation_data=(X_test, y_test))

# Load best model model.load_weights('model-tgs-
salt.h5')

# Evaluate on validation set (this must be equals to
the best log_loss) model.evaluate(X_test, y_test,
verbose=1)

# Predict on train, val and test
preds_train = model.predict(X_train, verbose=1)
preds_test =  model.predict(X_test, verbose=1)




# Threshold predictions
preds_train_t =  (preds_train  >
0.5).astype(np.uint8) preds_test_t =  (preds_test  >
0.5).astype(np.uint8)
```

A.3 Angle characterization

```
import numpy as np import pandas as pd
import matplotlib.pyplot as plt from matplotlib import
cm import gdal
from scipy import signal
from skimage.transform import (hough_line,
hough_line_peaks, probabilistic_hough_line)

# read img file into an array
geo =
gdal.Open('./USGS_NED_13_n33w092_IMG/USGS_NED_13_n33w0
92_IMG.img') arr = geo.ReadAsArray()
```

```
x, y  =  np.arange(np.shape(arr)[0]),
np.arange(np.shape(arr)[0])  xv, yv = np.meshgrid(x,y)

# subdivide entire DEM along with the corresponding
meshgrid values x_sub1   = xv[0:7000, 7000:-1]
y_sub1    =   yv[0:7000,  7000:-1]
dem_sub1 = arr[0:7000, 7000:-1]

# subdivide this region into a smaller region of
interest x_sub2     = x_sub1[2000:4000, 1500:3500]
y_sub2    =  y_sub1[2000:4000, 1500:3500]
dem_sub2 = dem_sub1[2000:4000, 1500:3500]

# subdivide to isolate a ridge-swale region along the
levee x_sub3   = x_sub2[1350:1650, 1400:1700]



y_sub3    = y_sub2[1350:1650, 1400:1700]
dem_sub3 = dem_sub2[1350:1650, 1400:1700]

# Split region into six sections along axis 0
dem_split0 = np.split(dem_sub3, 6, 0)

# loop over the split section to split along the other
axis   for split0,i in
zip(dem_split0,range(3)):#len(subArrays))):

# split the region into six sections along axis 1
dem_split1 = np.split(split0, 6, 1)
for split1,j in
zip(dem_split1,range(4)):#(len(subArrays2))): ###
Implement processing steps
# take the gradient
gradx    =     np.gradient(split1,axis=0) grady     =
np.gradient(split1,axis=1) grad = np.sqrt(gradx**2 +
grady**2) gradp90 = np.sqrt(gradx**2 + grady**2)
```

```
# set gradient values below  the  90th  percentile  to
0  p90 = np.percentile(np.abs(grad), 90)
gradp90[np.abs(grad) < p90] = 0

# apply a median filter to the 90th percentile of
gradients  medp90 =
signal.medfilt2d(gradp90,kernel_size=3)

### Apply the hough filter # example can be found at
h, theta, d = hough_line(medp90) rows, cols =
medp90.shape
_, angle, dist = hough_line_peaks(h, theta, d,
num_peaks=1) y0 = (dist - 0 * np.cos(angle)) /
np.sin(angle)
y1  =  (dist - cols *  np.cos(angle)) / np.sin(angle)
# Create a plot of the results



plt.figure(figsize=(10,9))

plt.subplot(2,2,1,aspect='equal')
plt.pcolormesh(medp90, cmap=plt.cm.gray)
plt.xlabel('Distance [m]',fontsize=25)
plt.ylabel('Distance [m]',fontsize=25) plt.title('(a)
Filtered Data',fontsize=27)

plt.subplot(2,2,(3,4)) plt.imshow(np.log(1 + h),
extent=[np.rad2deg(theta[-1]), np.rad2deg(theta[0]),
d[-1], d[0]],
cmap=plt.cm.gray)#, aspect=1/1.5) plt.title('(c) Hough
transform',fontsize=27) plt.xlabel('Angles
[degrees]',fontsize=25) plt.ylabel('Distance
[pixels]',fontsize=25)

plt.subplot(2,2,2,aspect='equal')
plt.pcolormesh(split1,cmap=cm.terrain,vmin=28,
vmax=32) plt.plot((0, cols), (y0, y1),'-
r',linewidth=3.5) plt.ylim([0,50])
```

```
plt.title('(b) Trendline',fontsize=27) cbar =
plt.colorbar()
cbar.set_label('Elevation [m]',fontsize=23)
plt.xlabel('Distance [m]',fontsize=25)
plt.ylabel('Distance [m]',fontsize=25)


plt.rc('xtick',  labelsize=23)     # fontsize of the
tick labels plt.rc('ytick',  labelsize=23)   #
fontsize of the tick labels

plt.tight_layout() plt.show()

xy_vals = []

# Split region into six sections along axis 0



n_split = 6 # number of splits along each axis
dem_split0 = np.split(dem_sub3, n_split, 0)
x0 = np.split(x_sub3,  n_split,  0)  y0 =
np.split(y_sub3,  n_split, 0)

# loop over the split section to split along the other
axis for i in range(len(dem_split0)):

# split the region into six sections along axis 1
dem_split1 = np.split(dem_split0[i],  n_split,  1) x1
= np.split(x0[i], n_split, 1)
y1 = np.split(y0[i], n_split, 1) for j in
range(len(dem_split1)):
### Implement processing steps # take the gradient
split1 = dem_split1[j]
gradx    =     np.gradient(split1,axis=0) grady     =
np.gradient(split1,axis=1) grad = np.sqrt(gradx**2 +
grady**2) gradp90 = np.sqrt(gradx**2 + grady**2)
```

```
# set gradient values below  the  90th  percentile  to
0  p90 = np.percentile(np.abs(grad), 90)
gradp90[np.abs(grad) < p90] = 0

# apply a median filter to the 90th percentile of
gradients  medp90 =
signal.medfilt2d(gradp90,kernel_size=3)

x = x1[j] x_0 = x[0] y = y1[j]

### Hough Filter stuff
h, theta, d = hough_line(medp90)
_, angle, dist = hough_line_peaks(h, theta, d,
num_peaks=1)




x_ = np.arange(np.shape(split1)[0])
y_ = (dist - x_*np.cos(angle))/np.sin(angle) +
np.min(y)

# limit lines to only fall within bounds of
corresponding sub-image  if np.min(y) > np.min(y_):
x_0 = x_0[y_ >= np.min(y)]  y_ = y_[y_ >= np.min(y)]

if np.max(y) < np.max(y_):
x_0 = x_0[y_ <=  np.max(y)] y_ = y_[y_ <= np.max(y)]

xy_vals.append((x_0, y_))

# plot plt.figure(figsize=(12,10))
plt.gca().set_aspect('equal')
plt.pcolormesh(x_sub3, y_sub3,
dem_sub3,cmap=cm.terrain,vmin=28, vmax=32)

plt.rc('xtick', labelsize=26) # fontsize of the tick
labels plt.rc('ytick', labelsize=26) # fontsize of the
tick labels cbar = plt.colorbar()
cbar.set_label('Elevation [m]',fontsize=26)
```

```
plt.suptitle('Ridge-Swale Region along the Mississippi
River',fontsize=30) plt.xlabel('Pixels',fontsize=28)
plt.ylabel('Pixels',fontsize=28)

for xy in xy_vals:
plt.plot(xy[0], xy[1],'-r',linewidth=3.5)

plt.locator_params(axis='x', nbins=5) plt.show()
```

Unit Conversion Factors

Multiply	By	To Obtain
degrees (angle)	0.01745329	radians
feet	0.3048	meters
miles (U.S. statute)	1,609.347	meters
square miles	2.589998 E+06	square meters

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
April 2021	Final	

4. TITLE AND SUBTITLE

Automated Characterization of Ridge-Swale Patterns Along the Mississippi River

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Alicia D. Downard, Stephen N. Semmens, and Bryant A. Robbins

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Geotechnical and Structures Laboratory
U.S. Army Engineer Research and Development Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER

ERDC/GSL TR-21-13

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Corps of Engineers
Washington, DC 20314-1000

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

Flood & Coastal Civil Works Direct Program, Remote Monitoring & Sensing; Critical Infrastructure Protection and Resilience Program, Office of Homeland Security (Funding Account Code U4375173, AMSCO Code 031398)

14. ABSTRACT

The orientation of constructed levee embankments relative to alluvial swales is a useful measure for identifying regions susceptible to backward erosion piping (BEP). This research was conducted to create an automated, efficient process to classify patterns and orientations of swales within the Lower Mississippi Valley (LMV) to support levee risk assessments. Two machine learning algorithms are used to train the classification models: a convolutional neural network and a U-net. The resulting workflow can identify linear topographic features but is unable to reliably differentiate swales from other features, such as the levee structure and riverbanks. Further tuning of training data or manual identification of regions of interest could yield significantly better results. The workflow also provides an orientation to each linear feature to support subsequent analyses of position relative to levee alignments. While the individual models fall short of immediate applicability, the procedure provides a feasible, automated scheme to assist in swale classification and characterization within mature alluvial valley systems similar to LMV.

15. SUBJECT TERMS

Geomorphology
Ridge and swale topography
Mississippi River
Flood control
Artificial intelligence
Backward erosion piping
Sand boils
Levees—Erosion
Geology
Alluvial plains
Alluvium
Alluvial fans
Digital elevation models
Machine learning

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	SAR	45	